CFD EXPERTS

Simulate the Future

WWW.CFDEXPERTS.NET

©2021 ANSYS, Inc.
All Rights Reserved.
Unauthorized use, distribution or duplication is prohibited.

Methods for Multi-Disciplinary Optimization and Robustness Analysis

ANSYS, Inc.
Southpointe
2600 Ansys Drive
Canonsburg, PA 15317
ansysinfo@ansys.com
http://www.ansys.com
(T) 724-746-3304
(F) 724-514-9494

Release 2021 R2
July 2021

ANSYS, Inc. and Ansys Europe, Ltd. are UL registered ISO 9001: 2015 companies.

Copyright and Trademark Information

© 2021 ANSYS, Inc. Unauthorized use, distribution or duplication is prohibited.

Ansys, Ansys Workbench, AUTODYN, CFX, FLUENT and any and all ANSYS, Inc. brand, product, service and feature names, logos and slogans are registered trademarks or trademarks of ANSYS, Inc. or its subsidiaries located in the United States or other countries. ICEM CFD is a trademark used by ANSYS, Inc. under license. CFX is a trademark of Sony Corporation in Japan. All other brand, product, service and feature names or trademarks are the property of their respective owners. FLEXlm and FLEXnet are trademarks of Flexera Software LLC.

Disclaimer Notice

THIS ANSYS SOFTWARE PRODUCT AND PROGRAM DOCUMENTATION INCLUDE TRADE SECRETS AND ARE CONFIDENTIAL AND PROPRIETARY PRODUCTS OF ANSYS, INC., ITS SUBSIDIARIES, OR LICENSORS. The software products and documentation are furnished by ANSYS, Inc., its subsidiaries, or affiliates under a software license agreement that contains provisions concerning non-disclosure, copying, length and nature of use, compliance with exporting laws, warranties, disclaimers, limitations of liability, and remedies, and other provisions. The software products and documentation may be used, disclosed, transferred, or copied only in accordance with the terms and conditions of that software license agreement.

ANSYS, Inc. and Ansys Europe, Ltd. are UL registered ISO 9001: 2015 companies.

U.S. Government Rights

For U.S. Government users, except as specifically granted by the ANSYS, Inc. software license agreement, the use, duplication, or disclosure by the United States Government is subject to restrictions stated in the ANSYS, Inc. software license agreement and FAR 12.212 (for non-DOD licenses).

Third-Party Software

See the legal information in the product help files for the complete Legal Notice for Ansys proprietary software and third-party software. If you are unable to access the Legal Notice, contact ANSYS, Inc.

Published in the U.S.A.

Table of Contents

Release 2021 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information
of ANSYS, Inc. and its subsidiaries and affiliates.

List of Figures

Release 2021 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

List of Tables

Release 2021 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information
of ANSYS, Inc. and its subsidiaries and affiliates.

Chapter 1: Introduction

Optimization and robustness analysis have become important tools for the virtual development of industrial products. In parametric optimization, the optimization variables are systematically modified by mathematical algorithms in order to get an improvement of an existing design or to find a global optimum. The design variables are defined by their lower and upper bounds or by several possible discrete values. In real world industrial optimization problems, the number of design variables can often be very large. Unfortunately, the efficiency of mathematical optimization algorithms decreases with increasing number of design variables. For this reason, several methods are limited to a moderate number of variables, such as gradient based and Adaptive Response Surface Methods. With the help of sensitivity analysis the designer identifies the variables which contribute most to a possible improvement of the optimization goal. Based on this identification, the number of design variables may be dramatically reduced and an efficient optimization can be performed. Additional to the information regarding important variables, sensitivity analysis may help to decide, if the optimization problem is formulated appropriately and if the numerical CAE solver behaves as expected.

A modern approach to search for better designs or to compute the ”best” design has to introduce all available engineering know how and has to automate a multidisciplinary optimization process. By specifying the design criteria as objectives and constraints and specifying the space of all possible designs with optimization parameters, a framework for numerical optimization can be defined. Part of the challenge of defining a multidisciplinary optimization problem will be the communication of different design groups about conflicting objectives, fixed criteria or weighted com-promise objective functions. Consequently, the degree of non-linearity has to be taken into account in the optimization process. Because of that the resulting optimization problem may become very noisy, very sensitive to design changes or ill conditioned for mathematical function analysis (non-differentiable, non-convex, non-smooth).

That defines the requirements to a modern optimization tool. Arbitrary solvers have to be connected, optimization strategies for smooth as well as for non-smooth or even ill-posed problems have to be available.

Besides the problem to find an optimal design, the evaluation of the robustness, for example, the sensitivity of unavoidable scatter of design variables due to the structural response, becomes more and more important. Very often, optimized designs tend to be very sensitive to small (sometimes random) fluctuations of parameters. Such phenomena may occur due to system instabilities like bifurcation problems in the structure. Design robustness can be checked by applying a systematic perturbation analysis based on a randomly generated design sample set. Statistics on the sample set allows the evaluation of robustness of the optimized design.

1.1. optiSLang Documentation Set

The following guides form the optiSLang documentation set.

- optiSLang and Statistics on Structures Beta Features

Describes the beta features contained in the optiSLang and Statistics on Structures software.

- optiSLang Excel Add-in Guide

  Describes how to use and troubleshoot the optiSLang add-in for Microsoft Excel.

- optiSLang Interfaces and Customization Guide

  Describes the available optiSLang external and programming interfaces. It also provides links to the available API documentation.

- optiSLang Inside Workbench

  Describes how to use the Ansys Workbench optiSLang Extension.

- optiSLang Installation and Licensing Guide

  Describes how to install optiSLang and configure licensing options using the optiSLang standalone installer. For installation instructions using the Ansys unified installer, see the Ansys, Inc. Installation Guides.

- optiSLang Python API Guide

  Provides an overview of the optiSLang Python API which is used to create, modify, and customize optiSLang projects and functionality with Python scripting.

- optiSLang Tutorials

  Demonstrates how to use optiSLang to solve different types of problems.

- optiSLang User's Guide

  Describes how to use and configure the optiSLang software.

- optiSLang Web Service API Documentation

  Provides an overview of the optiSLang Web Service REST API which is used to interact with the optiSLang Web Service back end.

- optiSLang Web Service Installation and Configuration Guide

  Describes how to install and configure the optiSLang Web Service platform.

- optiSLang Web Service User's Guide

  Describes how to use the optiSLang Web Service platform.

- Statistics on Structures Tutorials

  Demonstrates how to use SoS to solve different types of problems.

- Statistics on Structures User's Guide

  Describes how to install and use the SoS software.

Release 2021 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

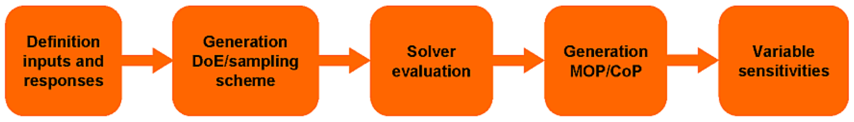
Chapter 2: Sensitivity Analysis

By definition, sensitivity analysis is the study of how the uncertainty in the output of a model can be apportioned, qualitatively or quantitatively, to different sources of variation in the input of a model (Saltelli et al. 2008 (p. 90)). In order to quantify this contribution, variance based methods are very suitable. With these methods the proportion of the output variance, which is caused by an random input variable, is directly quantified.

Variance based sensitivity analysis is also very suitable as an optimization pre-processing tool. By representing continuous optimization variables by uniform distributions without variable interactions, variance based sensitivity analysis quantifies the contribution of the optimization variables to a possible improvement of the model responses. In contrast to local derivative based sensitivity methods, the variance based approach quantifies the contribution with respect to the defined variable ranges.

Unfortunately, sufficiently accurate variance based methods require huge numerical effort due to the large number of simulation runs. Therefore, often meta-models are used to represent the model responses by surrogate functions in terms of the model inputs. However, many meta-model approaches are available and it is often not clear which one is most suitable for which problem (Roos et al. 2007 (p. 90)). Another disadvantage of meta-modeling is its limitation to a small number of input variables. Due to the so-called "curse of dimensionality" the approximation quality decreases for all meta-model types dramatically with increasing dimension. As a result, an enormous number of samples is necessary to represent high-dimensional problems with sufficient accuracy. In order to overcome these problems, we developed the Metamodel of Optimal Prognosis (Most and Will 2008 (p. 90)). In this approach the optimal input variable subspace together with the optimal meta-model are determined with help of an objective and model independent quality measure, the Coefficient of Prognosis. In the following chapter the necessity of such a procedure is explained by discussing other existing methods for sensitivity analysis. After presenting the MOP concept in detail, the strength of this approach is clarified by a comparison with very common meta-model approaches such as Kriging and neural networks

Figure 2.1: Flowchart of optiSLang Sensitivity Analysis

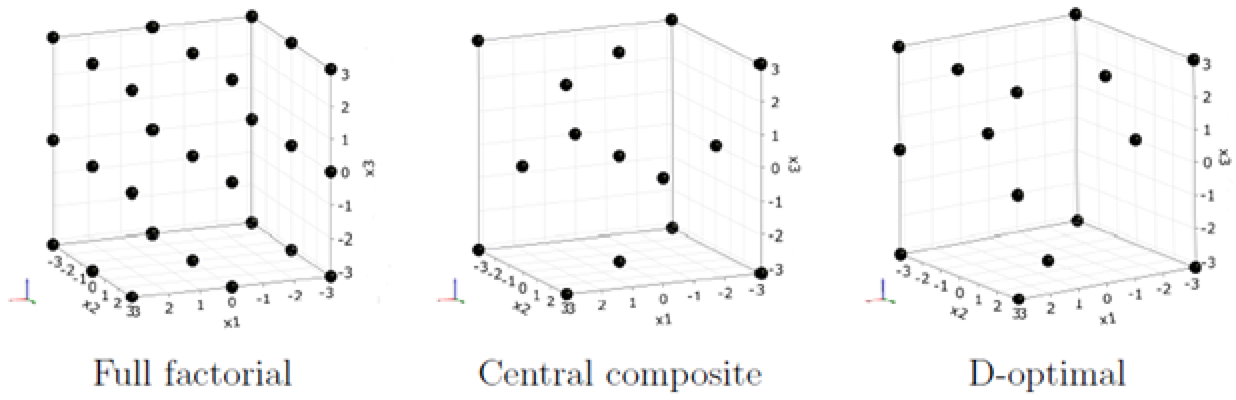

Definition inputs and responses
Generation DoE/sampling scheme
Solver evaluation
Generation MOP/CoP
Variable sensitivities

In Figure 2.1: Flowchart of optiSLang Sensitivity Analysis (p. 11) the general flow of an optiSLang sensitivity analyis is shown: after the definition of the design variables and model responses the design space is scanned by Design of Experiments or sampling methods. These samples are evaluated by the solver and for each design the model responses are determined. Approximation models are built for each model response and assessed regarding their quality. Finally variance based sensitivity indices are estimated by using the approximation models.

2.1. Scanning the Space of Input Variables

In order to perform a global sensitivity analysis, the space of the design variables has to be scanned by discrete realizations. Each realization is one set of values belonging to the specified inputs. As design exploration often deterministic Designs of Experiments (DoE) are applied (Myers and Montgomery 2002 (p. 90)). These design schemes are mainly based on a regular arrangement of the samples, as in the full factorial design. Generally the number of samples increases exponentially with increasing dimension. Fractional factorial designs use only a part of the full factorial samples, however the number of levels in each direction is limited. In Figure 2.2: Classical Design of Experiment Schemes (p. 12) a few typical DoE schemes are shown.

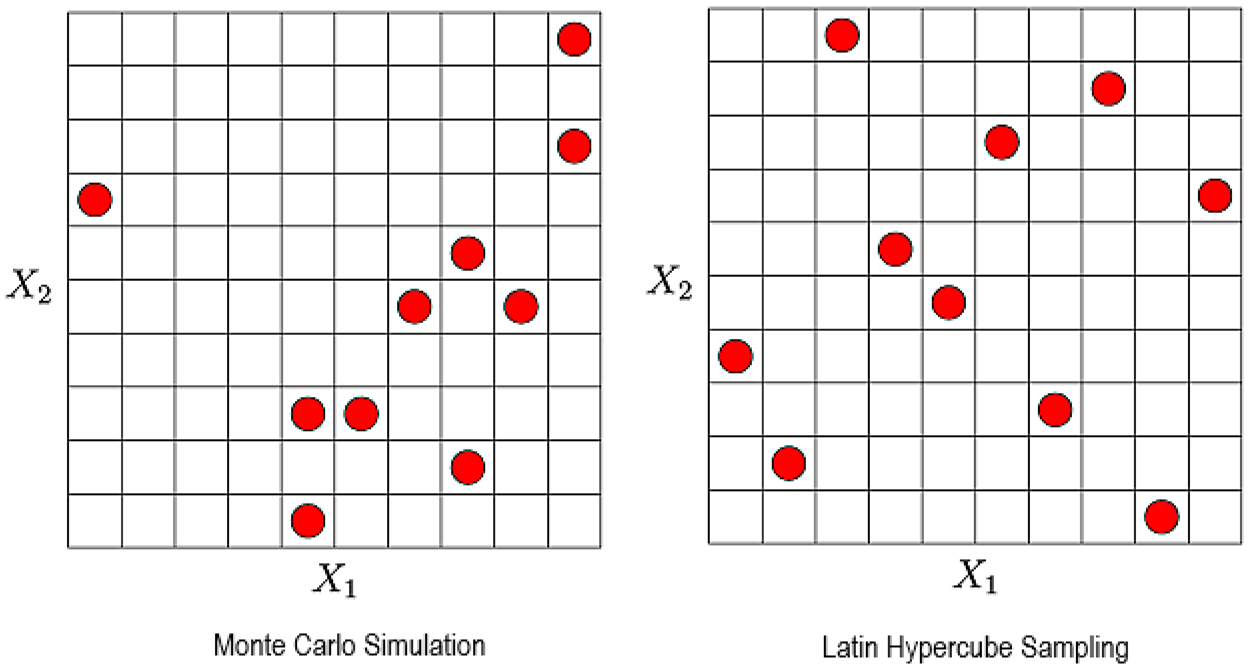
Figure 2.2: Classical Design of Experiment Schemes



The central composite design contains, additionally to the 2^k full factorial design points, the midpoints of each hypercube surface. D-optimal designs are generated in order to represent a linear or quadratic function with optimal accuracy. Further details about classical DoE schemes, which are also available in optiSLang, can be found in (Myers and Montgomery 2002 (p. 90)).

As alternatives to deterministic design exploration methods, stochastic sampling schemes can be applied. A very common approach is random sampling, which is called Monte Carlo Simulation (MCS). For design exploration the design variables are assumed to follow a uniform distribution with given lower and upper bounds. The random samples are generated independent in the given design space. If only a small number of samples is used, often clusters and holes can be observed in the MCS sampling set as shown in Figure 2.3: Stochastic Sampling Schemes (p. 13).

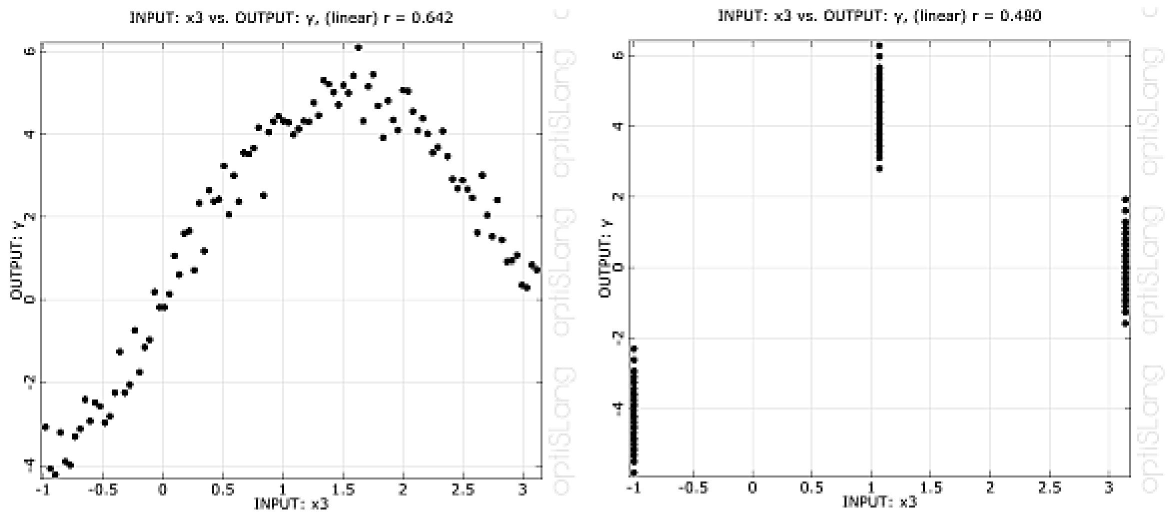
Figure 2.3: Stochastic Sampling Schemes



More critical is the appearance of undesired correlations between the input variables. These correlations may have a significant influence on the estimated sensitivity measures. In order to overcome such problems, optiSLang provides optimized Latin Hypercube Sampling (LHS) (McKay, Beckman, and Conover 1979 (p. 90)), where the input distributions and the specified input correlations are represented very accurately even for a small number of samples. In the standard Latin Hypercube approach the minimization of the undesired correlation is performed by using the method according to (Iman and Conover 1982 (p. 89)). Furthermore an advanced Latin Hypecube Sampling (ALHS) is available, where the correlation errors are minimized by stochastic evolution strategies (Hungtington and Lyrintzis 1998 (p. 89)). This approach is recommended if the number of input variables is less than 50.

Compared to Latin Hypercube sampling deterministic design schemes have two main disadvantages: They are limited to a small number of variables due to the rapidly increasing number of required samples when increasing the model dimension. Further, a reduction of the number of inputs does not improve the information gained from the samples, since only two or three levels are used in each dimension. This is illustrated in Figure 2.4: Dimension Reduction (p. 14). In this figure, a nonlinear function having one major and four almost unimportant variables is evaluated. Using the LHS, the nonlinearity can be represented very well in the reduced space. In the case of the full factorial design, which contains three levels in each directions, again only three positions are left in the reduced space and the dimension reduction does not allow a better representation of the model response.

Figure 2.4: Dimension Reduction



Dimension reduction for a nonlinear function with five inputs based on Latin Hypercube Sampling (left) with 100 samples and full factorial design (right) with 3^5 = 243 samples

2.2. Variance Based Sensititvity Analysis

See the following sections:

2.2.1. First Order and Total Effect Sensitivity Indices

Assuming a model with a scalar output *Y* as a function of a given set of *m* random input parameters X_i

$$Y=f(X_1,X_2,\ldots,X_m), \tag{2.1}$$

the first order sensitivity index is defined as

$$S_i=\frac{V(Y|X_i)}{V(Y)}, \tag{2.2}$$

where *V(Y)* is the unconditional variance of the model output and $V(Y|X_i)$ is the variance of *Y* caused by a variation of X_i only.

Since first order sensitivity indices measure only the decoupled influence of each variable an extension for higher order coupling terms is necessary. Therefore total effect sensitivity indices have been introduced

$$S_{Ti}=1-\frac{V(Y|X_{\sim i})}{V(Y)}, \tag{2.3}$$

where $V(Y|X_{\sim i})$ is the variance of *Y* caused by all model inputs without X_i .

In order to estimate the first order and total sensitivity indices, a matrix combination approach is very common (Saltelli et al. 2008 (p. 90)). This approach calculates the conditional variance for each variable with a new sampling set. In order to obtain a certain accuracy, this procedure requires often more than 1000 samples for each estimated conditional variance. Thus, for models with a large number of variables and time consuming solver calls, this approach can not be applied efficiently.

2.2.2. Coefficient of Correlation

The coefficient of correlation is the standardized covariance between two random variables *X* and *Y*

$$\rho(X,Y)=\frac{COV(X,Y)}{\sigma_X\sigma_Y}, \tag{2.4}$$

where *COV(X, Y)* is the covariance and σ is the standard deviation. This quantity, known as the linear correlation coefficient, measures the strength and the direction of a linear relationship between two variables. It can be estimated from a given sampling set as follows

$$p(X,Y)\approx\frac{1}{N-1}\frac{\sum_{i=1}^{N}(x_i-\hat{\mu}_X)(y_i-\hat{\mu}_Y)}{\hat{\sigma}_X\hat{\sigma}_Y} \tag{2.5}$$

where *N* is the number of samples, x_i and y_i are the sample values, and $\hat{\mu}_X$ and $\hat{\sigma}_X$ are the estimates of the mean value and the standard deviation, respectively. The estimated correlation coefficient becomes more inaccurate, as its value is closer to zero, which may cause a wrong deselection of apparently unimportant variables.

If both variables have a strong positive correlation, the correlation coefficient is close to one. For a strong negative correlation ρ is close to minus one. The squared correlation coefficient can be interpreted as the first order sensitivity index by assuming a linear dependence. The drawback of the linear correlation coefficient is its assumption of just a linear dependence. Based on the estimated coefficients only, it is not possible to decide on the validity of this assumption. Correlation coefficients, which assume a higher order dependence or use rank transformations solve this problem only partially. Additionally, often interactions between the input variables are important. These interactions can not be quantified with the linear and higher order correlation coefficients.

We can summarize that although the correlation coefficient can be simply estimated from a single sampling set, it can only quantify first order effects with an assumed dependence without any quality control of this assumption.

2.3. Polynomial Based Sensitivity Analysis

See the following sections:

2.3.1. Polynomial Regression

A commonly used approximation method is polynomial regression, where the model response is generally approximated by a polynomial basis function of linear or quadratic order with or without coupling terms. The model output y_i for a given set **x** $_i$ of the input parameters **X** can be formulated as the sum of the approximated value $\hat{y}_i$ and an error term ϵ_i.

$$y(\mathrm{x}_i)=\hat{y}_i(\mathrm{x}_i)+\epsilon_i=\mathrm{p}^T(\mathrm{x}_i)\beta+\epsilon_i, \tag{2.6}$$

where **p(x)** is the polynomial basis,

$$\mathrm{p}^T(\mathrm{x})=[\,1\ x_1x_2x_3\ldots x_1^2x_2^2x_3^2\ldots x_1x_2\ x_1x_3\ldots x_2x_3\ldots\,], \tag{2.7}$$

and β is a vector containing the unknown regression coefficients. These coefficients are generally estimated from a given set of sampled support points by assuming independent errors with equal variance at each point. By using a matrix notation the resulting least squares solution reads

$$\hat{\beta}=(\mathbf{P}^T\mathbf{P})^{-1}\mathbf{P}^T\mathbf{y}, \tag{2.8}$$

where **P** is a matrix containing the basis polynomials of the support point samples and **y** is the vector of support point values.

2.3.2. Coefficient of Determination

The Coefficient of Determination (CoD) can be used to assess the approximation quality of a polynomial regression model. This measure is defined as the relative amount of variation explained by the approximation (Montgomery and Runger 2003 (p. 90))

$$R^2=\frac{SS_R}{SS_T}=1-\frac{SS_E}{SS_T},\ 0\leq \mathrm{R}^2\leq 1 \tag{2.9}$$

where SS_T is equivalent to the total variation of the output Y, SS_R represents the variation due to the regression, and SS_E quantifies the unexplained variation,

$$SS_T=\sum_{i=1}^{N}(y_i-\mu_Y)^2,\ SS_R=\sum_{i=1}^{N}(\hat{y}_i-\mu_{\hat{Y}})^2,\ SS_E=\sum_{i=1}^{N}(y_i-\hat{y}_i)^2. \tag{2.10}$$

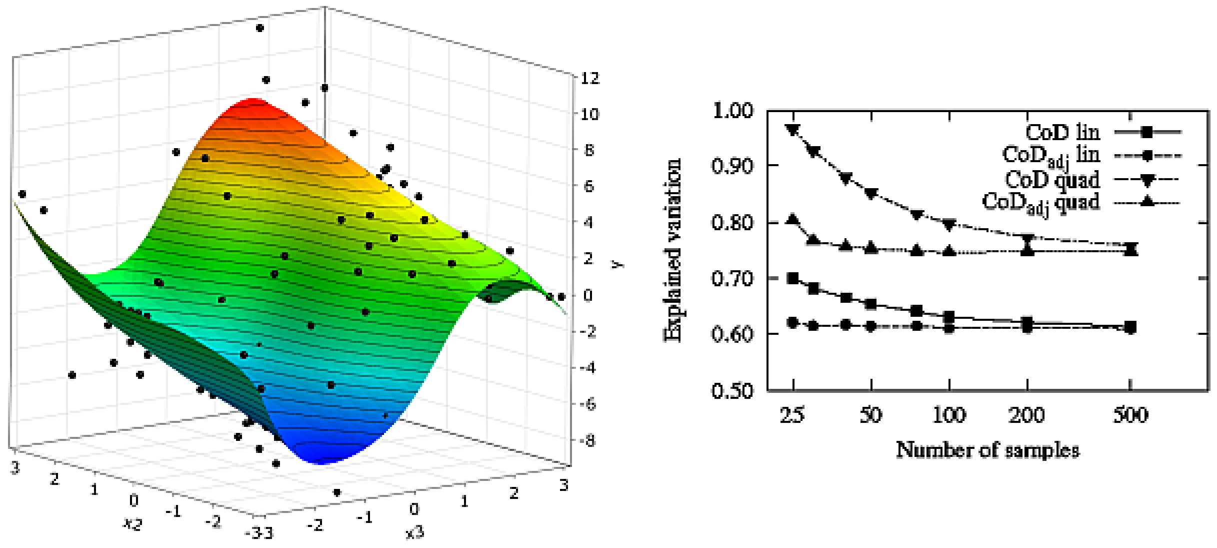
If the CoD is close to one, the polynomial approximation represents the support point values with small errors. However, the polynomial model would fit exactly through the support points, if their number is equivalent to the number of coefficients p. In this case, the CoD would be equal to one, independent of the true approximation quality. In order to penalize this over-fitting, the adjusted Coefficient of Determination was introduced (Montgomery and Runger 2003 (p. 90))

$$R^2_{adj}=1-\frac{N-1}{N-p}(1-R^2). \tag{2.11}$$

However, the over-estimation of the approximation quality cannot be avoided completely.

Release 2021 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 2.5: Subspace Plot and Convergence of the CoD Measures



Subspace plot of the investigated nonlinear function (Equation 2.12 (p. 17)) and convergence of the CoD measures with increasing number of support points.

In order to demonstrate this statement, an investigation of a nonlinear analytical function is performed. The function of five independent and uniformly distributed input variables reads as follows

$$Y=0.5X_1+X_2+0.5X_1X_2+5.0sin(X_3)+0.2X_4+0.1X_5, \quad -\pi\leq X_i\leq\pi, \tag{2.12}$$

where the contributions of the five inputs to the total variance are X_1: 18.0%, X_2: 30.6%, X_3: 64.3%, X_4: 0.7%, X_5: 0.2%. This means, that the three variables, X_1, X_2 and X_3, are the most important.

In Figure 2.5: Subspace Plot and Convergence of the CoD Measures (p. 17), the convergence of the standard CoD of linear and quadratic response surfaces is shown, where a strong over-estimation of the approximation quality can be noticed, when the number of samples is relatively small. Even the adjusted CoD shows a similar behavior. This fact limits the CoD to cases where a large number of support points compared to the number of polynomial coefficients is available. However, in industrial applications this is often not the case. Another disadvantage of the CoD measure is its limitation to polynomial regression. For other local approximation models, like interpolating Kriging, this measure may be equal or close to one, however the approximation quality may be poor.

2.3.3. Coefficient of Importance

The Coefficient of Importance (CoI) was developed to quantify the input variable importance by using the CoD measure. Based on a polynomial model, including all investigated variables, the CoI of a single variable Xi with respect to the response Y is defined as follows

$$CoI(X_i,Y)=CoI_{Y,X_i}=R^2_{Y,X}-R^2_{Y,X\sim i}, \tag{2.13}$$

where $R^2_{Y,X}$ is the CoD of the full model including all terms of the variables in **X** and $R^2_{Y,X\sim i}$ is the CoD of the reduced model, where all linear, quadratic and interactions terms belonging to X_i are removed from the polynomial basis. For both cases the same set of sampling points is used. If a variable has low importance, its CoI is close to zero, since the full and the reduced polynomial regression model have a similar quality. The CoI is equivalent to the explained variation with respect to a single input variable, since the CoD quantifies the explained variation of the polynomial approximation. Thus it is an estimate of the total effect sensitivity measure given in Equation 2.3 (p. 14). If the polynomial model contains important interaction terms, the sum of the CoI values should be larger than the CoD of the full model.

Since it is based on the CoD, the CoI is also limited to polynomial models. If the total explained variation is over-estimated by the CoD, the CoI may also give a wrong estimate of the variance contribution of the single variables. However, in contrast to the Coefficient of Correlation, the CoI can handle linear and quadratic dependencies including input variable interactions. Furthermore, an assessment of the suitability of the polynomial basis is possible. Nevertheless, an estimate of the CoI values using a full quadratic polynomial is often not possible because of the required large number of samples for high dimensional problems.

2.4. Metamodel of Optimal Prognosis

See the following sections:

2.4.1. Moving Least Squares Approximation

In the Moving Least Squares (MLS) approximation (Lancaster and Salkauskas 1981 (p. 89)) a local character of the regression is obtained by introducing position-dependent radial weighting functions. MLS approximation can be understood as an extension of the polynomial regression. Similarly the basis function can contain every type of function, but generally only linear and quadratic terms are used. The approximation function is defined as

$$\hat{y}(\mathbf{x})=\mathbf{p}^T(\mathbf{x})\mathbf{a}(\mathbf{x}), \tag{2.14}$$

with changing ("moving") coefficients **a(x)** in contrast to the constant global coefficients of the polynomial regression. The final approximation function reads

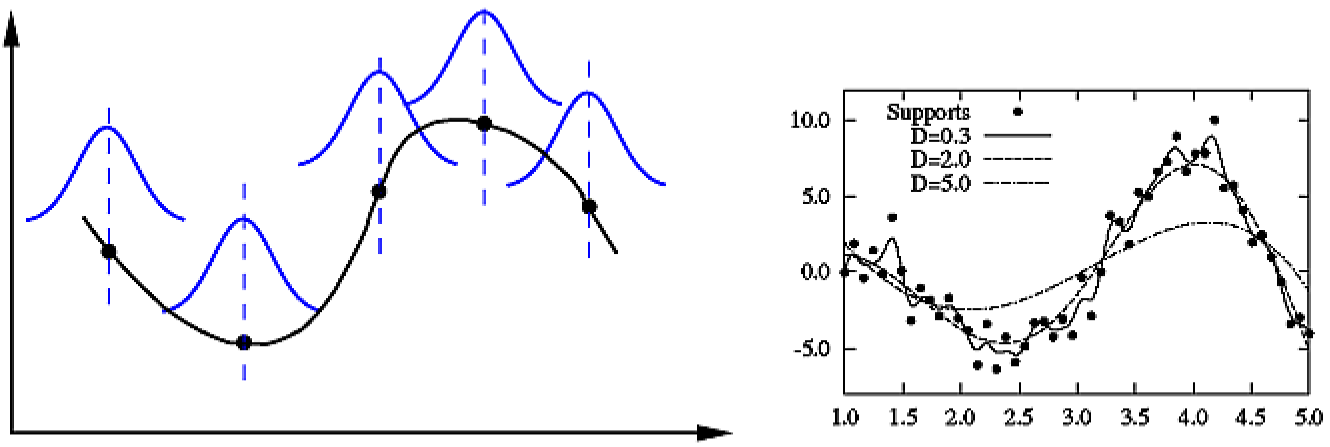
$$\hat{y}(\mathbf{x})=\mathbf{p}^T(\mathbf{x})\left(\mathbf{P}^T\mathbf{W}(\mathbf{x})\mathbf{P}\right)^{-1}\mathbf{P}^T\mathbf{W}(\mathbf{x})\mathbf{y}, \tag{2.15}$$

where the diagonal matrix **W(x)** contains the weighting function values corresponding to each support point. Distance-dependent weighting functions $w=w(||x-x_i||)$ have been introduced. Mostly the well known Gaussian weighting function is used

$$w_{\exp}(||x-x_i||)=\exp\left(-\frac{||X-X_i||^2}{\alpha^2 D^2}\right), \tag{2.16}$$

where the influence radius *D* directly influences the approximation error and α is a numerical constant. A suitable choice of this quantity enables an efficient smoothing of noisy data. In Figure 2.6: Local Weighting Principle and Smoothing Effect (p. 19) the local weighting principle and the smoothing effect is shown.

Figure 2.6: Local Weighting Principle and Smoothing Effect



Local weighting of support point values (left) and influence of the influence radius *D* on the smoothing of the MLS approximation function (right).

The MLS approach has the advantage that no training is necessary before an approximation point can be evaluated. At each point only the weighting factors and the local polynomial coefficients have to be calculated. This makes this method very fast compared to other approximation techniques.

2.4.2. Coefficient of Prognosis

In (Most and Will 2008 (p. 90)) a model independent measure to assess the model quality was proposed. This measure is the Coefficient of Prognosis (CoP), which is defined as follows

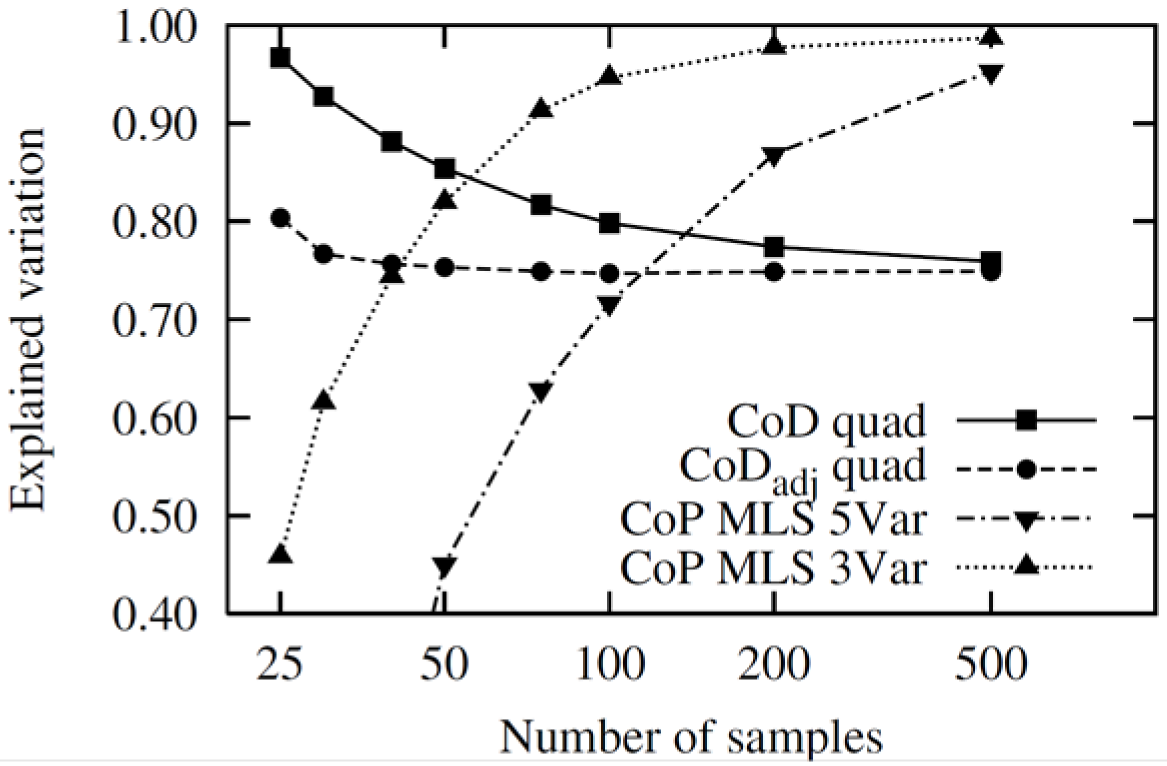
$$CoP = 1 - \frac{SS_E^{\mathrm{P}rediction}}{SS_T}, \tag{2.17}$$

where $SS_E^{\mathrm{P}rediction}$ is the sum of squared prediction errors. These errors are estimated based on cross validation. In the cross validation procedure, the set of support points is mapped to q subsets. Then the approximation model is built by removing subset *i* from the support points and approximating the subset model output $\tilde{y}i$ using the remaining point set. This means that the model quality is estimated only at those points which are not used to build the approximation model. Since the prediction error is used instead of the fit, this approach applies to regression and even interpolation models.

The evaluation of the cross validation subsets, which are usually between 5 and 10 sets, causes additional numerical effort in order to calculate the CoP. Nevertheless, for polynomial regression and Moving Least Squares, this additional effort is still quite small since no complex training algorithm is required. For other meta-modeling approaches such as neural networks, Kriging and even Support Vector Regression, the time consuming training algorithm has to be performed for every subset combination.

In Figure 2.7: Convergence of the CoP Measure by Using MLS Approximation Compared to the Polynomial CoD Measure (p. 20), the convergence of the CoP values for an MLS approximation of the nonlinear coupled function given in Equation 2.12 (p. 17) is shown in comparison to the polynomial CoD. The figure indicates that the CoP values are not over-estimating the approximation quality like the CoD does for a small number of samples.

Figure 2.7: Convergence of the CoP Measure by Using MLS Approximation Compared to the Polynomial CoD Measure

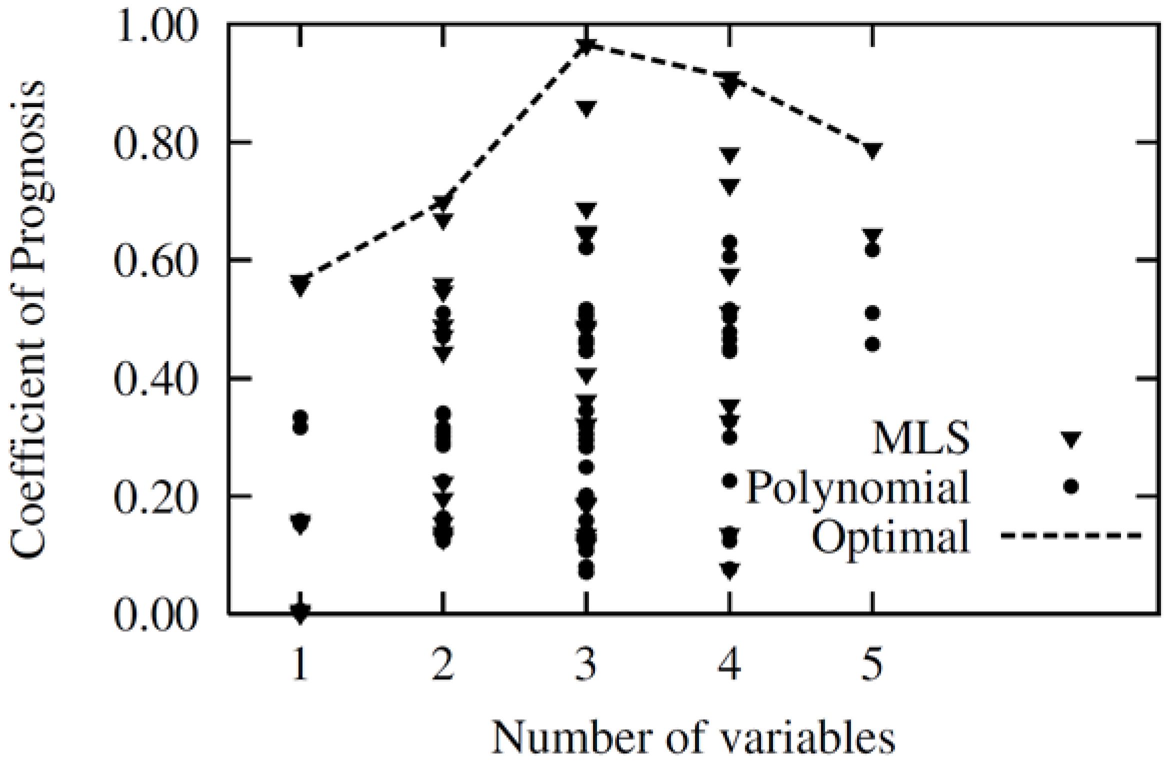


The influence radius of the MLS approximation is found by maximizing the CoP measure. As can be seen in Figure 2.7: Convergence of the CoP Measure by Using MLS Approximation Compared to the Polynomial CoD Measure (p. 20), the convergence of the approximation quality is much better if only the three most important variables are used in the approximation model.

2.4.3. Metamodel of Optimal Prognosis

As demonstrated in the previous section, the prediction quality of an approximation model may be improved if unimportant variables are removed from the model. This idea is adopted in the Metamodel of Optimal Prognosis (MOP) proposed in (Most and Will 2008 (p. 90)) which is based on the search for the optimal input variable set and the most appropriate approximation model (polynomial or MLS with linear or quadratic basis). Due to the model independence and objectivity of the CoP measure, it is well suited to compare the different models in the different subspaces.

Figure 2.8: CoP Values of Different Input Variable Combinations and Approximation Methods Obtained with the Analytical Nonlinear Function



In Figure 2.8: CoP Values of Different Input Variable Combinations and Approximation Methods Obtained with the Analytical Nonlinear Function (p. 21), the CoP values of all possible subspaces and all possible approximation models are shown for the analytical nonlinear function example. The figure indicates that there exists an optimal compromise between the available information, the support points and the model complexity i.e. the number of input variables. The MLS approximation by using only the three major important variables has a significantly higher CoP value than other combinations. However for more complex applications with many input variables it is necessary to test a huge number of approximation models. In order to decrease this effort, in (Most and Will 2008 (p. 90)) advanced filter technologies are proposed, which reduce the number of necessary model tests. Nevertheless, a large number of inputs requires a very fast and reliable construction of the approximation model. For this reason polynomials and MLS are preferred due to their fast evaluation.

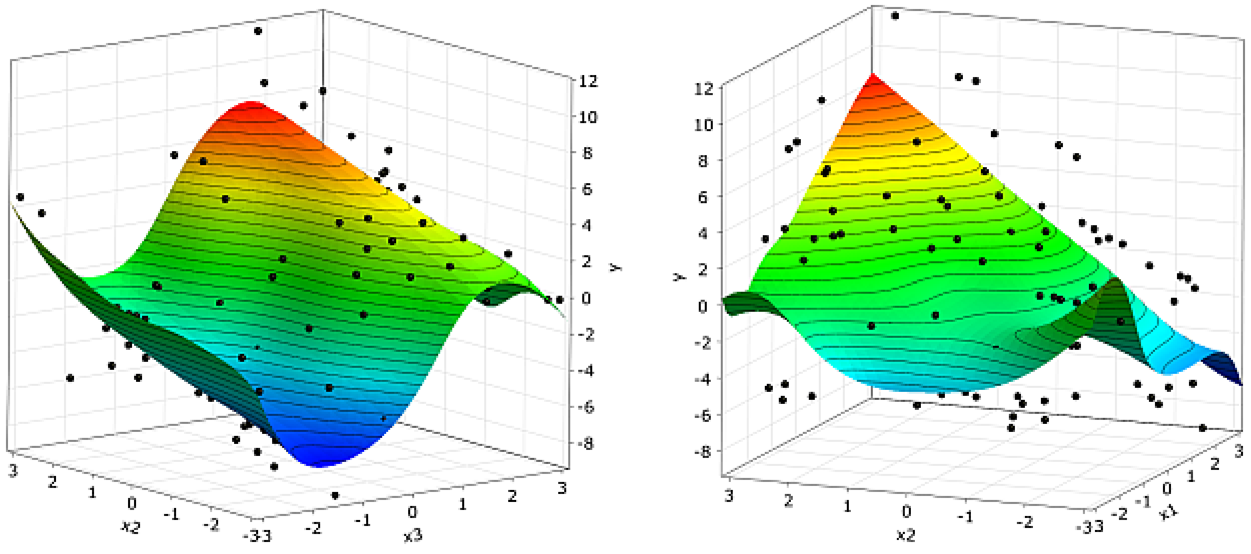
As a result of the MOP, we obtain an approximation model which contains the most important variables. Based on this meta-model, the total effect sensitivity indices, proposed in First Order and Total Effect Sensitivity Indices (p. 14), are used to quantify the variable importance. The variance contribution of a single input variable is quantified by the product of the CoP and the total effect sensitivity index estimated from the approximation model

$$CoP(X_i)=CoP\ .\ S_T^{MOP}(X_i). \tag{2.18}$$

Since interactions between the input variables can be represented by the MOP approach, they are considered automatically in the sensitivity indices. If the sum of the single indices is significantly larger as the total CoP value, such interaction terms have significant importance.

Additionally to the quantification of the variable importance, the MOP can be used to visualize the dependencies in 2D and 3D subspaces. This helps the designer to understand and to verify the solver model. In Figure 2.9: Subspace Plots (p. 22), two subspace plots are shown for the MOP of the analytical test function. In the X_2-X_3 and X_1- X_2 subspace plots, the sinusoidal function behavior and the coupling term can be observed. Additional parametric studies, such as global optimization can also be directly performed on the MOP. Nevertheless, a single solver run should be used to verify the final result of such a parametric study or optimization.

Figure 2.9: Subspace Plots

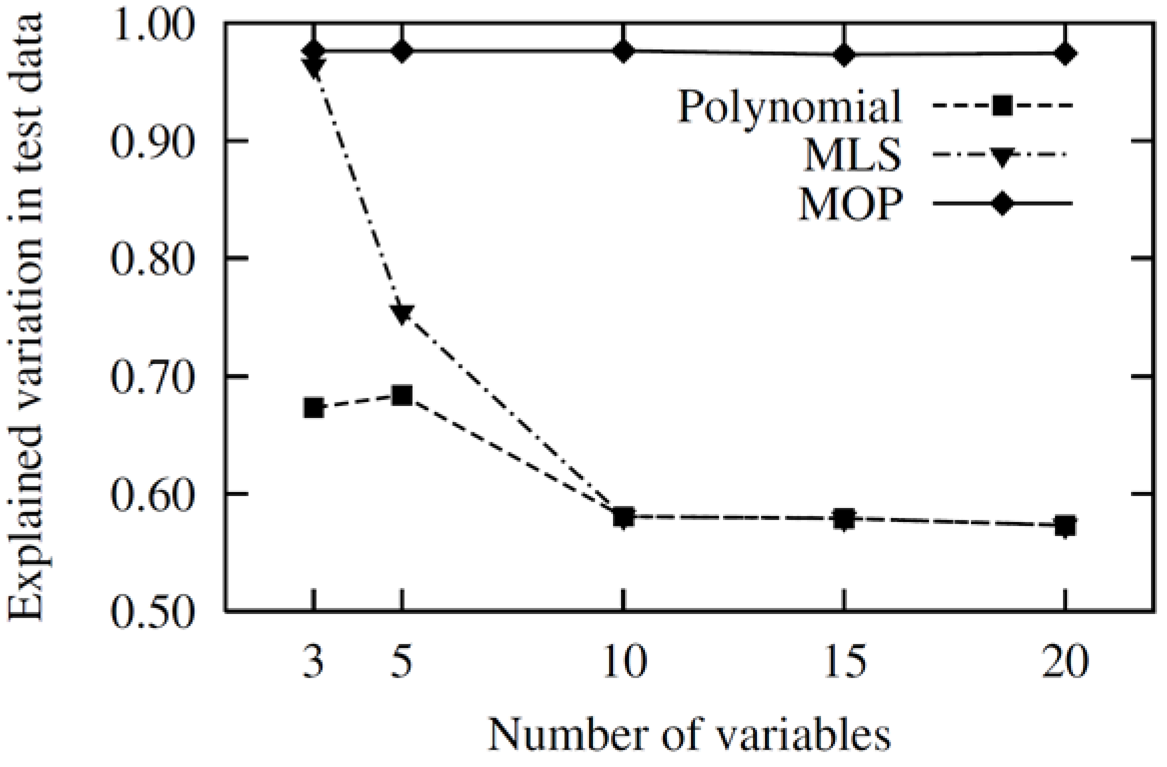


X_2-X_3 and X_1-X_2 subspace plots of the MOP of the nonlinear analytical function given in Equation 2.12 (p. 17).

2.5. Comparision With Other Approximation and Selection Methods

In this section we compare the MOP approach with other approximation and variable selection methods. Before using advanced meta-models, we investigate the test case with polynomials and Moving Least Squares. The analytical nonlinear function introduced in Coefficient of Determination (p. 16) is investigated by different numbers of input variables: only the three main important variables, all five variables and additional variables without any contribution. In Figure 2.10: Approximation Quality for Polynomial and MLS Approximation (p. 23), the explained variation of the polynomial and MLS approximation obtained with 100 support and 100 test points is shown with respect to the total number of variables. Due to the so-called "curse of dimensionality" the approximation quality decreases rapidly with increasing dimension. If the MOP is applied, the important variables are retained and the approximation is built in the optimal subspace. This leads to a high approximation quality even for larger input dimensions.

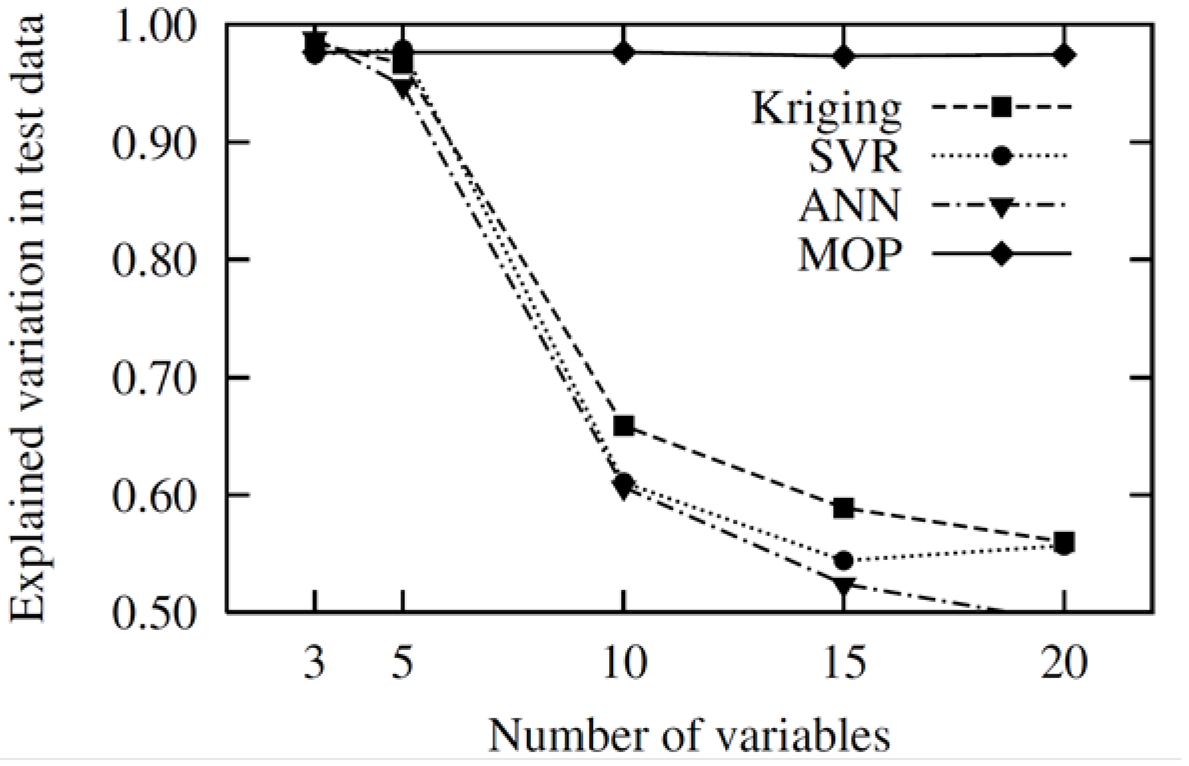
Figure 2.10: Approximation Quality for Polynomial and MLS Approximation



Approximation quality for polynomial and MLS approximation compared to the MOP approach for the analytical function with increasing number of input variables.

In the next step, we investigate the Kriging approximation, that is also known as Gaussian process model. In Figure 2.11, the approximation quality of the ordinary Kriging approach is shown for the test function. The significant decrease of the explained variation is similar to that of the MLS approach.

Figure 2.11: Approximation Quality for Kriging, Support Vector Regression (SVR) and Artificial Neural Networks (ANN)

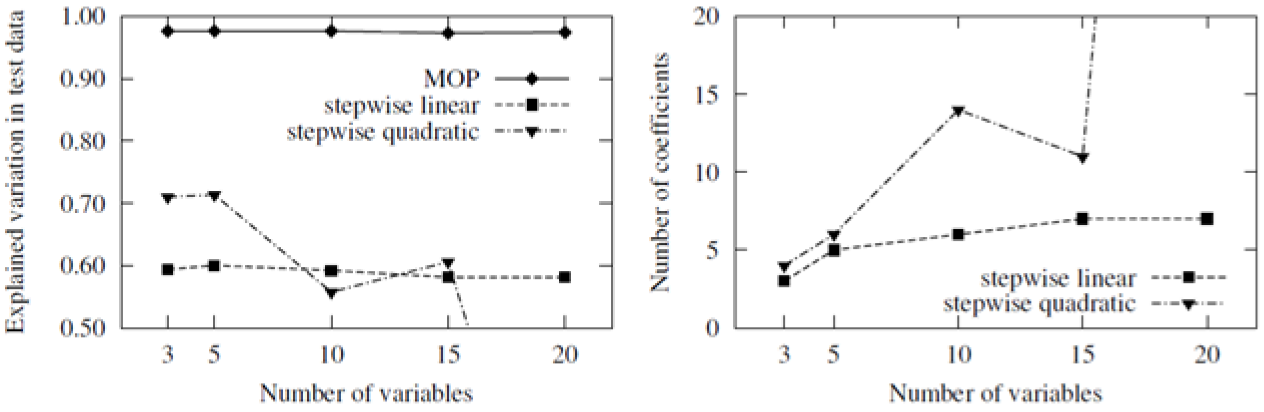


Approximation quality for Kriging, Support Vector Regression (SVR) and Artificial Neural Networks (ANN) compared to the MOP approach for the analytical function.

Furthermore, Support Vector Regression (SVR) and Artificial Neural Networks (ANN) are investigated. A detailed presentation of these methods can be found in (Roos et al. 2007 (p. 90)). The obtained results are also plotted in Figure 2.11: Approximation Quality for Kriging, Support Vector Regression (SVR) and Artificial Neural Networks (ANN) (p. 24), which exhibit a similar behavior as Kriging and MLS. All the presented results indicate that the utilization of complex meta-model approaches will not overcome the curse of dimensionality. However, the MOP enables the determination of the optimal variable sub-space by using fast and reliable approximation methods. In most cases this variable reduction leads to a significantly better approximation quality.

Finally the MOP approach is compared to the polynomial stepwise selection method. In this approach polynomial coefficients are selected by different importance criteria in order to detect the important variables. For comparison, the state of the art implementation in (MATLAB 2010 (p. 90)) is used. In this implementation important polynomial coefficients are selected by F-test statistics based on a given polynomial degree. The results given in Figure 2.12: Approximation Quality and Variable Selection of MATLAB's Stepwise Regression Approach Compared to the MOP Results (p. 25) indicate that the selection procedure works appropriately only for a linear polynomial basis. By using a full quadratic basis, the number of selected coefficients increases dramatically and the approximation quality decreases. This example clarifies the power of the prediction based variable selection applied inside the MOP approach.

Figure 2.12: Approximation Quality and Variable Selection of MATLAB's Stepwise Regression Approach Compared to the MOP Results



Release 2021 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information
of ANSYS, Inc. and its subsidiaries and affiliates.

Chapter 3: Multidisciplinary Optimization

See the following sections:

3.1. Single-Objective Optimization

In single-objective optimization problems the optimization task can be formulated by a single scalar-valued objective function

$$f(x_1,x_2,...,x_k)\rightarrow min, \tag{3.1}$$

which is often an implicit function of the design variables. The design variables can be defined as continuous variables with a lower and upper bound or as discrete variables which assume several discrete values. In unconstrained optimization prob- lems only the bounds or values of the design variables limit the optimization space. The optimizer searches between these limits for the minimum value of the objective function $f(x)$. For maximization problems the sign of the user defined objective function is inverted in optiSLang to get a minimization task.

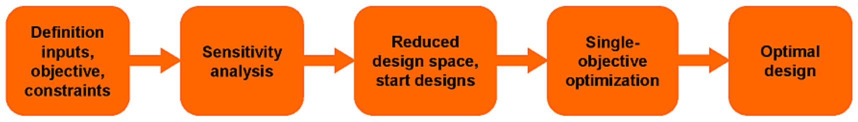
In engineering problems often additional restrictions have to be fulfilled by the optimal design. With help of equality and inequality constraints

$$\begin{aligned} g_i(x_1,x_2,...,x_k)&=0, i=1...m_e,\\ h_j(x_1,x_2,...,x_k)&\geq 0, j=1...m_u, \end{aligned} \tag{3.2}$$

such restrictions can be formulated. The constraint functions can represent limitations depending only on the input variables but also depending on all available model responses and any mathematical combination of both.

Equality constraints are very difficult to treat in industrial applications. Therefore, it is not allowed to define equality constraints in optiSLang. However, equality constraints fulfilled with a given tolerance can be formulated easily as inequality constraints and in that way they can be handled by all optimization methods available in optiSLang. In order to get an optimal convergence of the optimizer and sufficiently accurate fulfillment of the constraint conditions, it is very useful to scale the objective and the constraint functions to be in a similar range. This can be realized e.g. by using the results of the design exploration.

Figure 3.1: Recommended Flowchart for Single-Objective Optimization

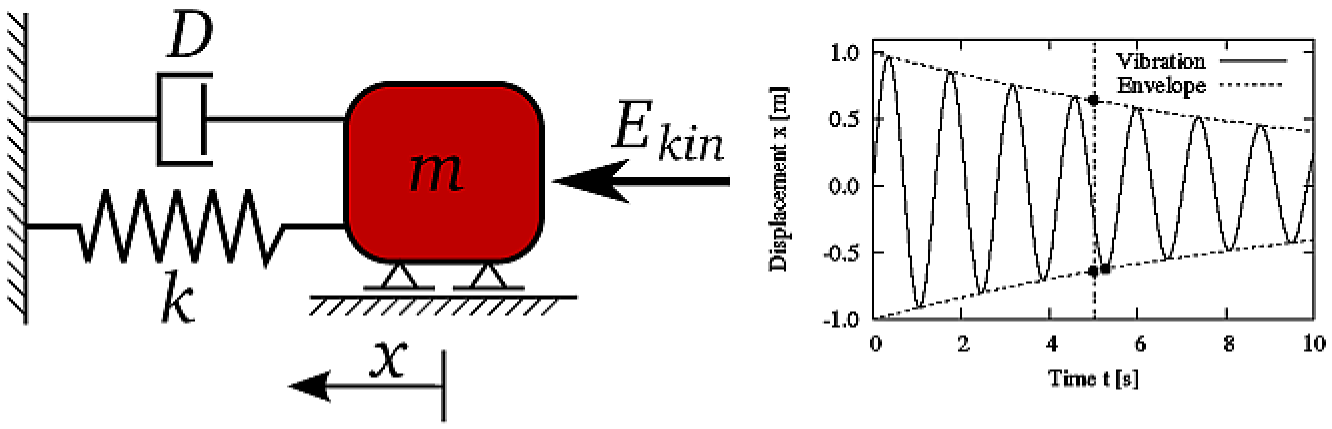

Definition inputs, objective, constraints
Sensitivity analysis
Reduced design space, start designs
Single-objective optimization
Optimal design

In Figure 3.1: Recommended Flowchart for Single-Objective Optimization (p. 27), the recommended flow of single-objective optimization procedure is shown: after the definition of the design variables and objective and constraint functions the design space is explored by sensitivity analysis. The obtained variable sensitivities may help to reduce the number of design variables. The best designs found in the sensitivity analysis could be used as start designs for the following optimization procedure which finally will determine an optimal design.

Accompanying Example: Optimization of a Damped Oscillator

The principles of the optimization methods described in the following sections are demonstrated by a simple accompanying example. For this purpose the maximum amplitude of a single degree-of-freedom damped oscillator shown in Figure 3.2: Damped Oscillator: System Properties and Oscillation Behavior (p. 28) shall be minimized.

Figure 3.2: Damped Oscillator: System Properties and Oscillation Behavior



The equation of motion can be formulated depending on the mass *m*, the spring stiffness *k* and damping ratio *D* as follows

$$\ddot{x}+2D\omega_0\dot{x}+\omega_0^2x=0, \tag{3.3}$$

where $\omega_0=\sqrt{k/m}$, is the undamped eigen-frequency. Assuming an initial kinetic energy E_{kin} and an initial position $x_0=0$, the time-dependent displacement function can be derived as follows

$$x(t)=e^{-D\omega_0 t}\sqrt{\frac{2E_{kin}}{m}}\,\frac{1}{\omega}\sin(\omega t), \tag{3.4}$$

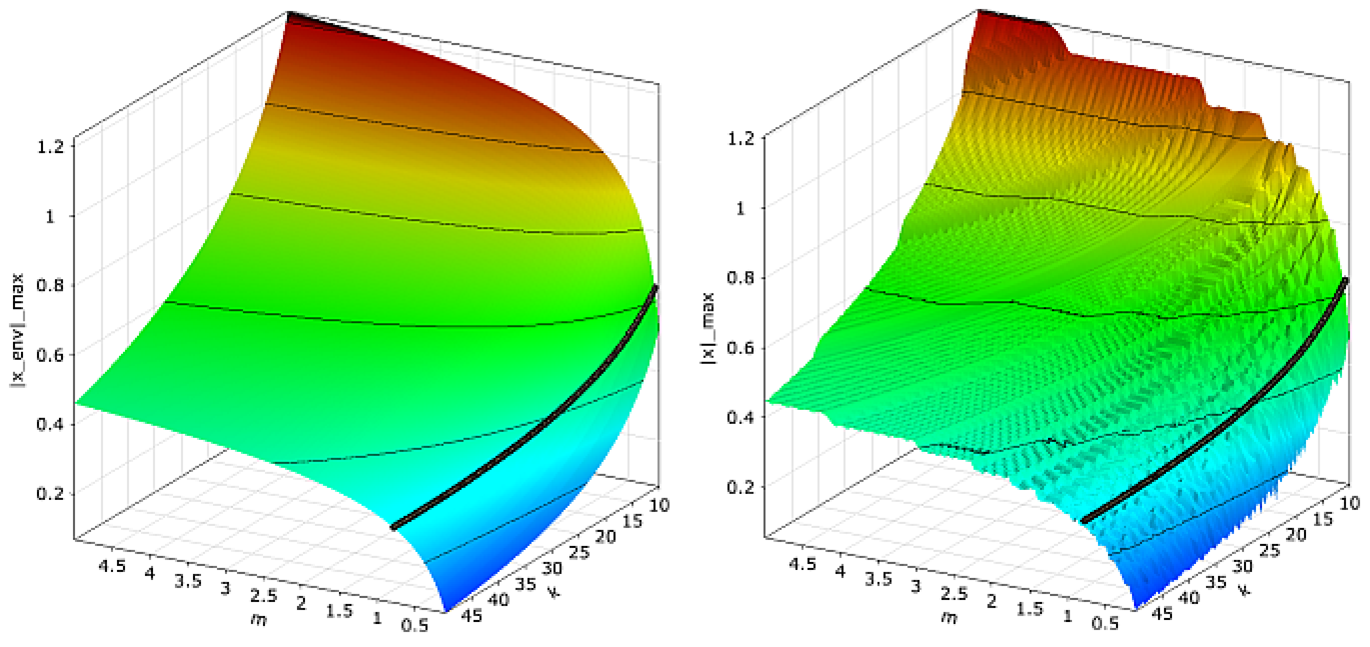
where $w=w_0\sqrt{1-D^2}$ is the damped eigen-frequency. The optimization goal is to minimize the maximum amplitude after 5 seconds by assuming *m* and *k* as design parameters and *D* and E_{kin} as constants

$$\begin{aligned} &m\in[0.1,5.0kg], \qquad D=0.02, \\ &k\in[10,50\,N/\mathrm{m}],\ E_{kin}=10\,\mathrm{Nm}. \end{aligned} \tag{3.5}$$

As optimization constraint, the damped eigen-frequency shall be limited: $w \leq 8\ 1/s$.

If the maximum amplitude after 5 seconds is estimated from the envelope curve shown in Figure 3.2: Damped Oscillator: System Properties and Oscillation Behavior (p. 28), the objective function is a smooth and well defined function of the design parameters as shown in Figure 3.3: Objective Function of the Damped Oscillator (p. 29).

Figure 3.3: Objective Function of the Damped Oscillator



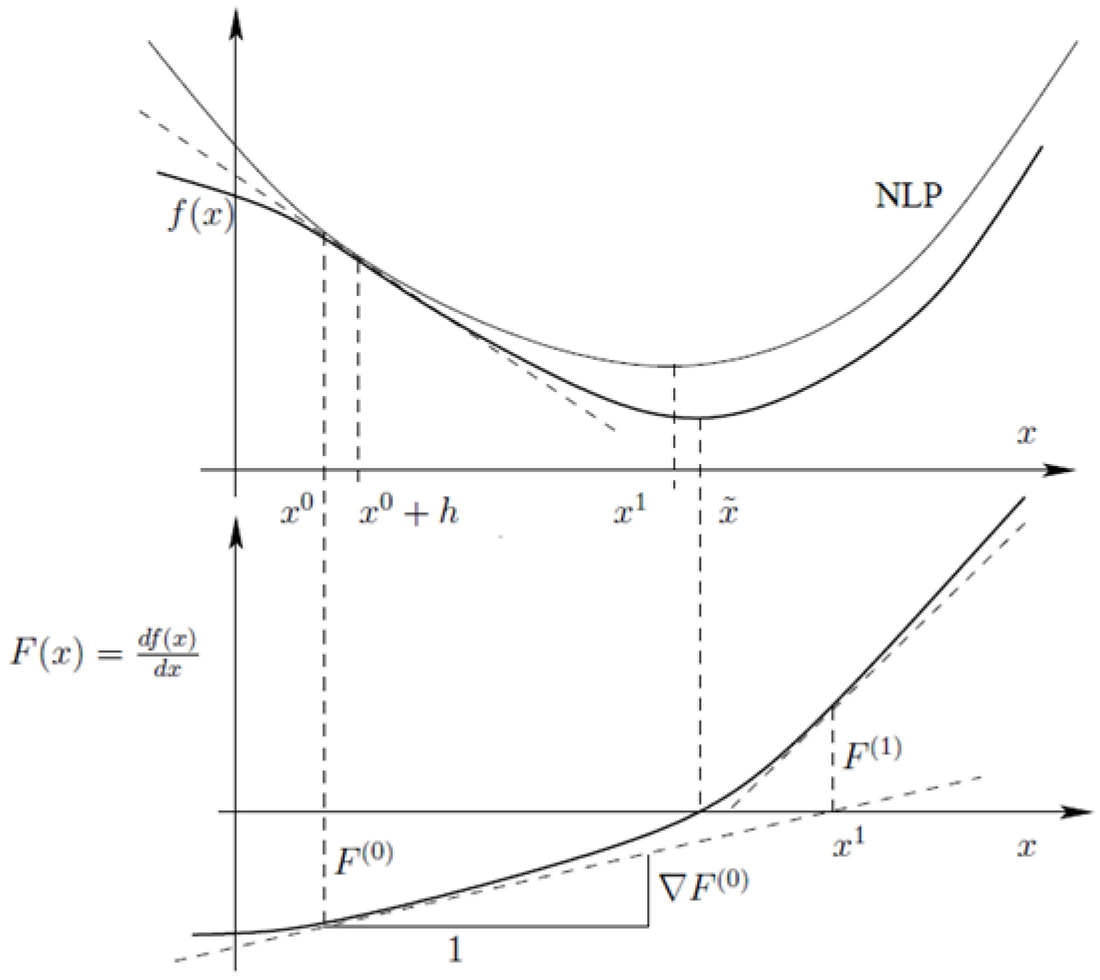
Objective function of the damped oscillator obtained by using an es- timate of the maximum amplitude from the envelope curve (left) and by using a coarse time discretization of the displacement curve (right).

The constraint condition is indicated in the figure by the function w = 8 1/s. If the maximum amplitude is obtained by a time discretization of Equation 3.4 (p. 28) using a coarse time interval, the resulting objective function may contain local oscillations, which may be interpreted as solver noise. In order to demonstrate the behavior of the different optimizers in the presence of small solver noise, a time step of 0.1 seconds is chosen, which leads to a slightly noisy objective function shown additionally in Figure 3.3: Objective Function of the Damped Oscillator (p. 29).

3.1.1. Gradient-Based Methods

Gradient-based optimization methods use local derivatives of the objective function to find the next local optimum. The minimum of a convex function can be determined by searching for the point, where the first derivative is equal to zero as shown in Figure 3.4: Iterative Search of a Gradient-Based Method Using a Quadratic Approximation of the Objective Function (p. 30).

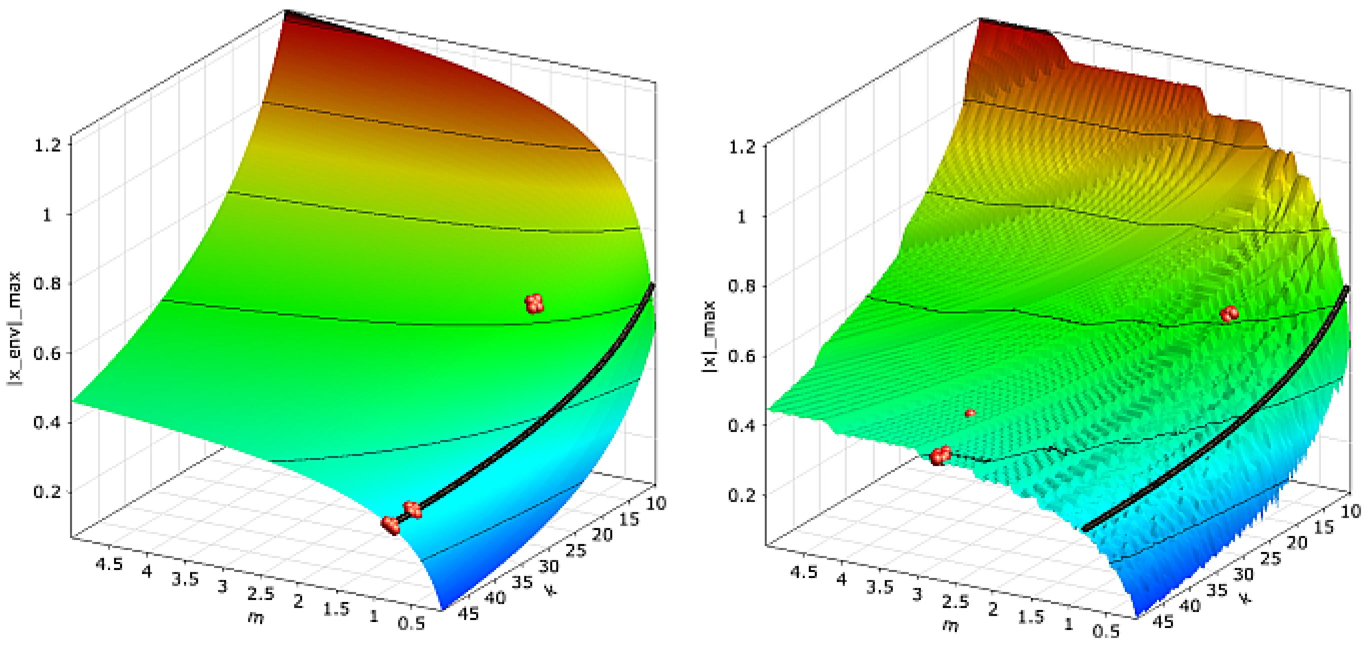
Figure 3.4: Iterative Search of a Gradient-Based Method Using a Quadratic Approximation of the Objective Function



If the second derivatives is known, the objective function can be approximated by a second order Taylor series. This procedure is called Nonlinear Programming (NLP). Optimization methods using first and second order derivatives are known as Newton methods (Kelley 1999 (p. 89)). If the CAE solver is treated as black box, the derivatives have to be calculated by numerical estimates. Since second order derivatives would require a large number of solver calls, full Newton methods are not applicable for complex optimization problems. For this reason quasi Newton methods have been developed, where the second order derivatives are approximated from the first order derivatives of previous iteration steps. Gradient-based methods distinguish each other mainly in the way how the second order derivatives are estimated and how additional constraint equations are considered. A good overview over different methods in given in (Kelley 1999 (p. 89)).

In optiSLang the NLPQL approach (Nonlinear Programming by Quadratic Lagrangian) is preferred (Schittkowski 1986 (p. 90)). First-order derivatives of the objective and constraint functions are estimated from central differences or by one-sided numerical derivatives using a specified differentiation interval. The NLPQL approach features efficient constraint handling using quadratic Lagrangian. Starting from a given start point, the method searches for the next local optimum and converges if the estimated gradients are below a specified tolerance. Since it is a local optimization method, it is recommended to use the best design of a global sensitivity analysis as start point in order to find the global optimum. The NLPQL is very efficient in low dimensions with up to 20 design variables. For higher dimensional problems the computation of the numerical derivatives becomes more and more expensive and other methods are more efficient. In the presence of noisy model responses the differentiation interval plays a crucial role. If its taken too small, the estimated gradient is distorted heavily by the solver noise and the NLPQL runs into a wrong direction. If the solver noise is not dominating the functional trend an increased differentiation interval could lead to a good convergence of the optimization procedure.

Figure 3.5: Convergence of the NLPQL Optimizer



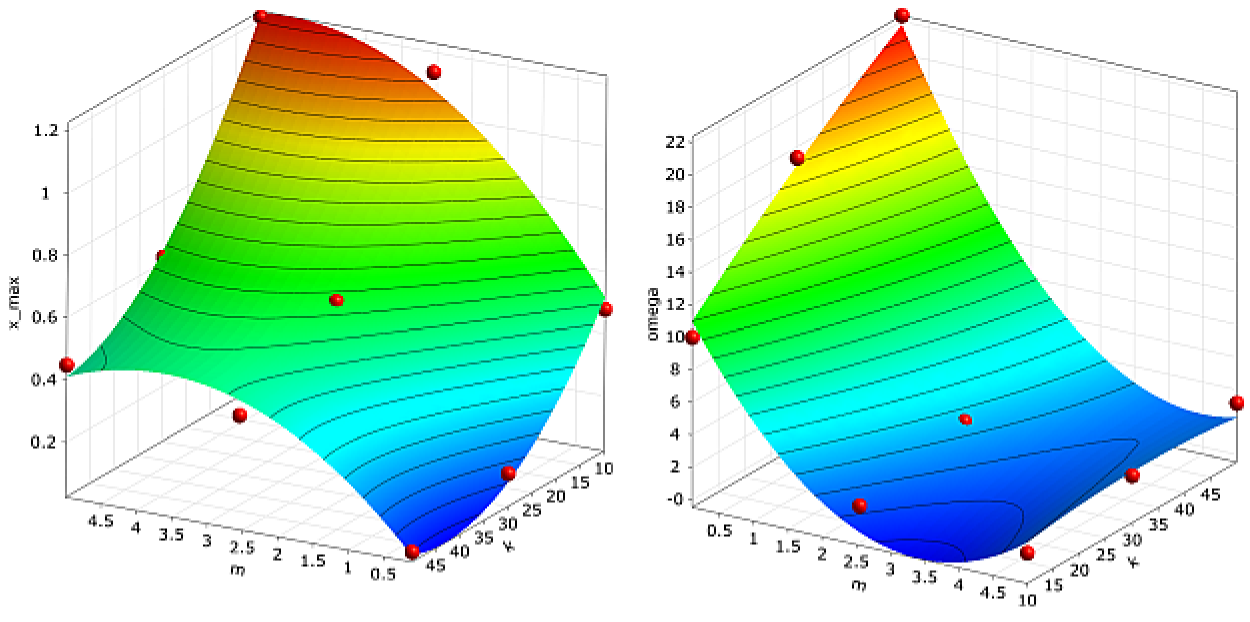
Convergence of the NLPQL optimizer for the oscillator optimization problem by using the smooth objective function based on the envelope estimate (left) and by using the noisy objective function from the time discretization (right).

In Figure 3.5: Convergence of the NLPQL Optimizer (p. 31), the convergence of the NLPQL approach is shown for the optimization of the damped oscillator: in the first investigation, the smooth objective function obtained from the envelope curve estimate is considered. In this case the optimizer converges in three iteration steps to the true optimum (m = 0.78 kg, k = 50.0 N/m). If the noisy objective function is investigated with a relatively small differentiation interval (1% of the design space) the optimizer runs into the wrong direction and will not converge to the true optimum. Nevertheless, by increasing the differentiation interval the convergence to the true optimum can be achieved for this example.

3.1.2. Response Surface Based Methods

Response surface methods replace the model responses by mathematical surrogate functions. On this base, the optimization problem is solved using the approximation model instead of time consuming solver calls. The classical procedure uses a Design of Experiments scheme which is evaluated by the solver. Based on these designs often polynomial functions are used for the approximation (Myers and Montgomery 2002 (p. 90)). However, the polynomial degree does often not represent the nonlinearity of the solver model and a low quality of the approximation leads often to unusable design suggestions. For this reason it is not recommended to use global polynomial response surface methods only to search for an optimal design.

Figure 3.6: Global Polynomial Approximation



Global polynomial approximation of the maximum amplitude (left, R^2_{adj} = 98%) and the damped eigen-frequency (right, R^2_{adj} = 96%) of the optimized oscillator using a quadratic basis and a full factorial design scheme.

In order to demonstrate the weakness of global polynomial response surface approximation in combination with classical DoE schemes, the maximum amplitude and the damped eigen-frequency of the damped oscillator are approximated by a quadratic polynomial. Support points are generated using a three-level full factorial design. Although the approximation quality is indicated to be very well by means of the adjusted Coefficient of Determination, the optimum found on the approximation (m = 3.31 kg, k = 50.0 N/m) is far away from the true optimum.

Optimization using the Metamodel of Optimal Prognosis

Due to the power of the Metamodel of Optimal Prognosis procedure in finding an optimal variable subspace and an optimal approximation model for each investigated model response, it is strongly recommended to use the MOP approximation for a first optimization step instead of global polynomial models. The Coefficient of Prognosis gives a much more reliable estimate of the approximation quality than the Coefficient of Determination as explained in Coefficient of Prognosis (p. 19). If the quality of a certain model response, which is used in the objective or constraint functions, is low, the optimization procedure may not find a useful optimized design. In order to check the quality of the optimal design found on the MOP, optiSLang provides a verification of the best design where the solver outputs and the true objective and constraint values are calculated by a single solver call. Often the best design found by the MOP based optimization is better than the best design of the preceding sensitivity analysis. If this is the case, this design could be used as a start design for a further local search.

In the presence of many constraint conditions often the MOP based best design violates one or more of these conditions if the approximation quality is not perfect. This problem can be solved by adjusting

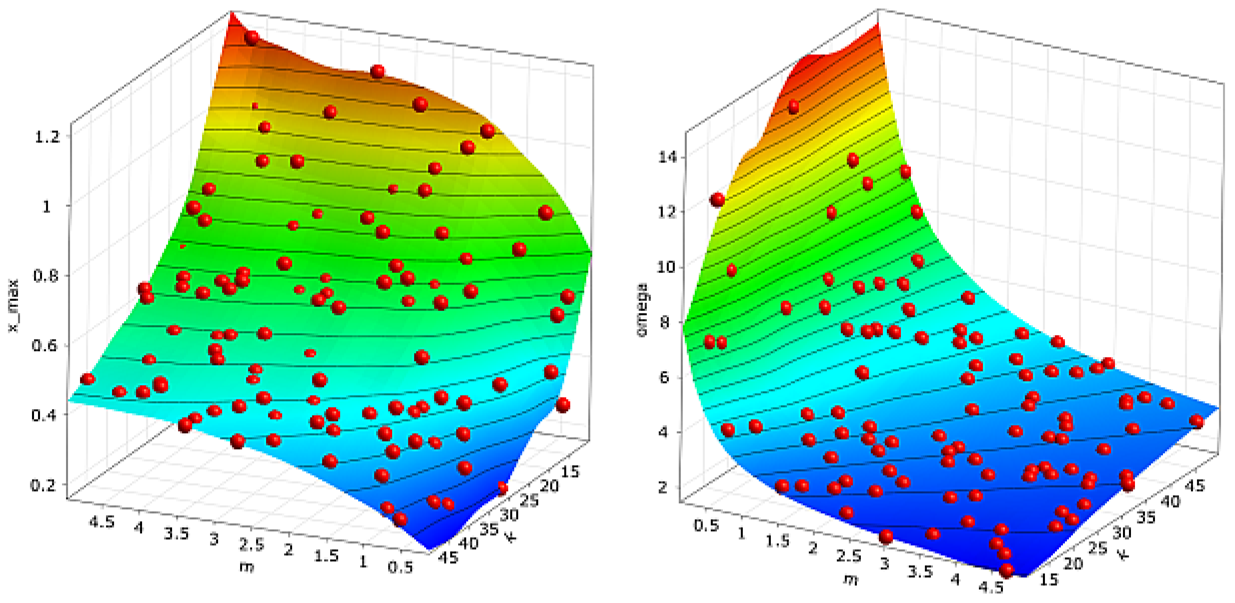
Release 2021 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

the limits of the corresponding constraint conditions in order to push the optimizer running on the approximation back to the feasible region. After each adjustment of the constraint conditions the best design should be verified to check for a possible fulfillment of the critical constraints.

Since the solver noise is smoothed by the MOP approximation, this procedure is more stable than the gradient-based methods. Furthermore, non-convex optimization problems can be solved by using global optimizers such as evolutionary algorithms (EA) on the approximation function. Nevertheless, generally the MOP uses uniformly distributed designs for the approximation model, which may lead to a insufficient local approximation quality around the optimum. Therefore, optimization using global approximation models should be understood as a low-cost pre-optimization step.

In order to demonstrate the benefit of an optimization on the MOP approximation, the damped oscillator is investigated by building the MOP with 100 Latin Hypercube samples. The approximation functions of the MOP are shown in Figure 3.7: Metamodel of Optimal Prognosis (p. 33).

Figure 3.7: Metamodel of Optimal Prognosis



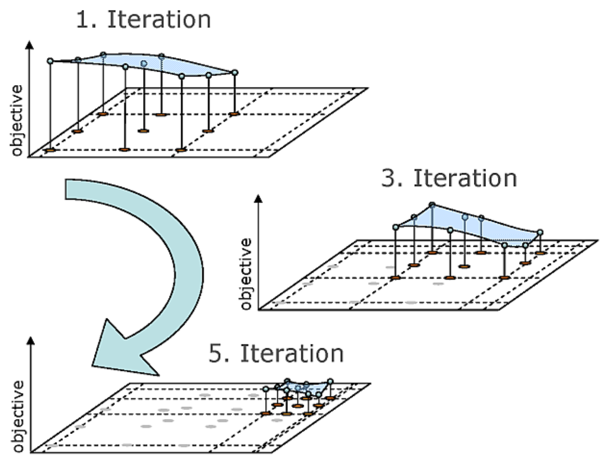
Approximation of the maximum amplitude (left, *CoP* = 99%) and of the damped eigen-frequency (right, *CoP* = 98%) by using the Metamodel of Optimal Prognosis with 100 Latin Hypercube samples

The figure indicates an excellent approximation quality in terms of the Coefficient of Prognosis. By using the MOP approximation functions the optimum is determined very close to the true optimum (m = 0.77 kg, k = 50.0 N/m). The approximated damped eigen-frequency at the obtained optimum is ω = 8.00 1/s, but the real eigen-frequency verified by the solver is ω = 8.04 1/s, which means that the constraint condition is slightly violated. In this case a reduction of the maximum allowed eigen-frequency would force the optimizer to stay in the feasible region.

Adaptive Response Surface Method

In order to improve the approximation quality around the optimum, adaptive meth- ods are very efficient. optiSLang provides a polynomial based local Adaptive Response Surface Method (ARSM). The ARSM procedure starts at a single start design, and an initial Design of Experiments (DoE) scheme is built having the start design as center point. For linear or quadratic polynomial models the corresponding D-optimal design schemes are preferred as DoE schemes. Based on the approximation of the model responses the optimal design is searched within the parameter bounds of the DoE scheme. In the next iteration step a new DoE scheme is built around this optimal design. Depending on the distance between the optimal designs of the current and previous iteration steps, the DoE scheme is moved, shrunken or expanded. This procedure is shown in principle in Figure 3.8. Further details about the adaptation procedure can be found in (Etman et al. 1996 (p. 89)) and (Stander and Graig 2002 (p. 90)).

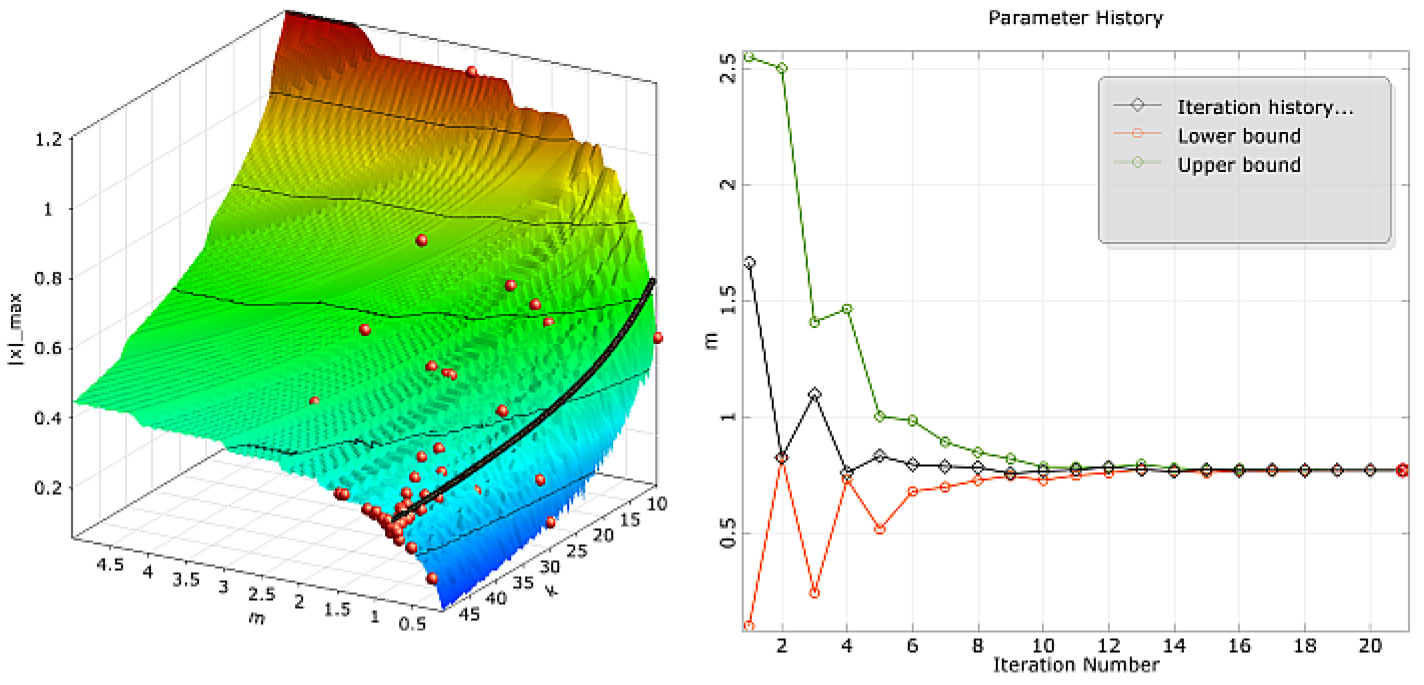
Figure 3.8: Adaptation of the Polynomial Approximation Scheme Inside the Adaptive Response Surface Method



The ARSM algorithm converges if the DoE is shrunken to a minimum size or if the change of the optimal design position and its objective value between two iteration steps is below a specified tolerance. Since the DoE scheme uses 50% more designs than needed for the polynomial approximation, the solver noise is smoothed and few failed designs are not problematic for the optimizer. Due to its efficiency for up to 20 variables and its robustness against solver noise, the ARSM is the method of choice for low dimensional single-objective optimization problems. It is recommended to use the optimal design of a preceding sensitivity analysis as start design for the ARSM. For a strongly localized search the start range of the initial DoE scheme should be reduced.

In Figure 3.9: Convergence of the Adaptive Response Surface Method (p. 35), the convergence of the ARSM optimizer is shown for the oscilla- tor problem by analyzing the noisy objective function.

Figure 3.9: Convergence of the Adaptive Response Surface Method



Convergence of the Adaptive Response Surface Method for the damped oscillator by using the noisy objective function: designs used for the adaptation (left) and modification of the local DoE bounds during the iteration (right).

By using a start range of 50% of the design space and a linear polynomial basis, the optimizer runs within a few iteration steps in the region of the true optimum. After 20 iteration steps the algorithm converges at m = 0.77 kg, k = 49.3 N/m, where the constraint condition is fulfilled. This example clarifies that the ARSM optimizer shows a stable convergence behavior for noisy model responses in contrast to the gradient based methods.

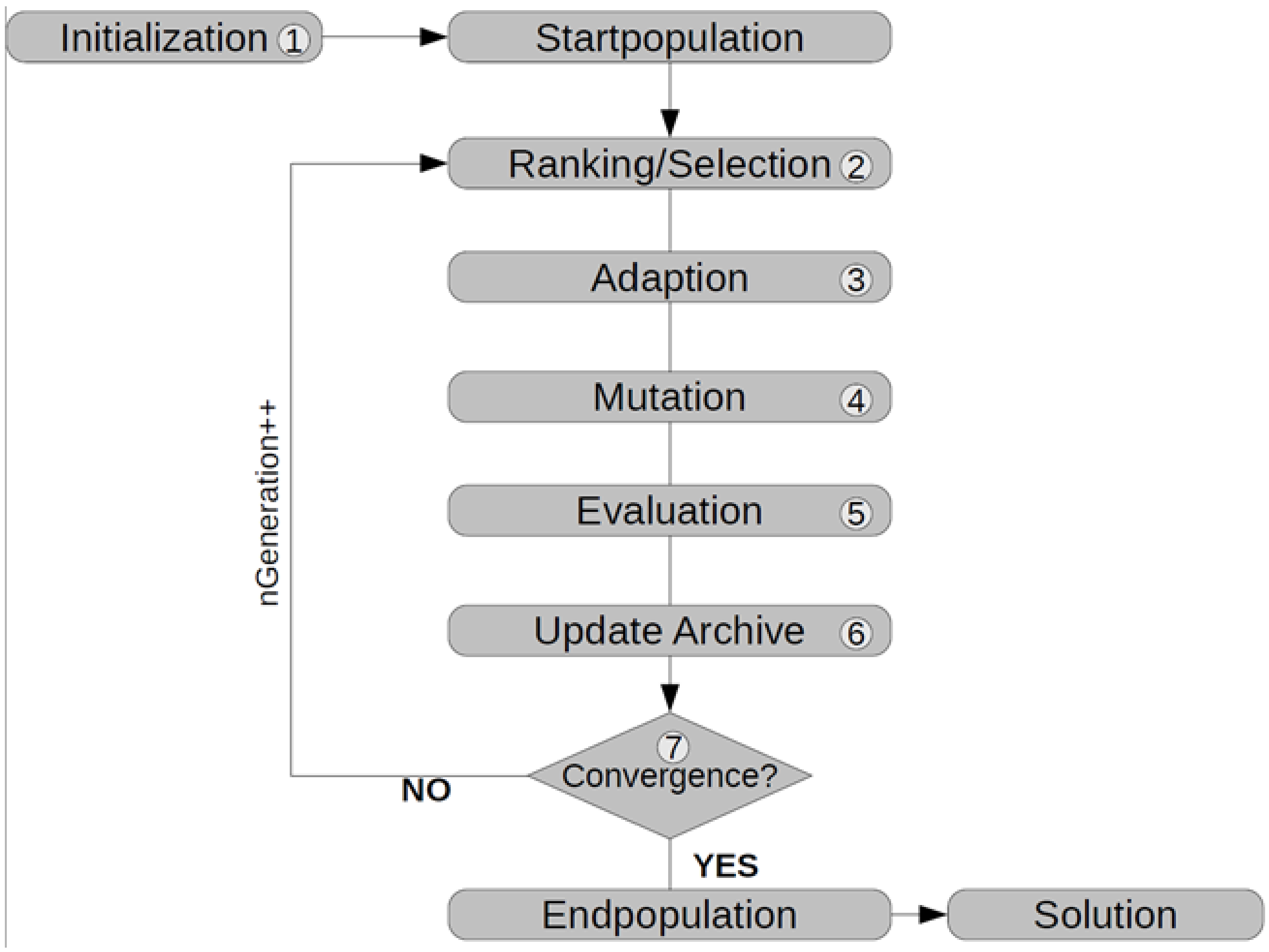
3.1.3. Population Based Methods

Population based optimization methods imitate natural processes like biological evolution or swarm intelligence. Based on the principle 'survival of the fittest' a population of artificial individuals searches the design space of possible solutions in order to find a better approximation for the solution of the optimization problem.

All kinds of population based algorithms are following a particular flow. It starts with a random initialization (see Figure 3.10: General Flowchart of Population Based Optimization Algorithms (p. 36), step 1) of a population containing μ individuals representing possible solutions to the given optimization problem.

After initialization the actual iteration loop starts, where every loop represents a generation. To create a new generation the fittest designs will be selected (step 2). The adaption referred to step 3 can be done in different ways, e.g. crossover or swarm movement. Mutation (step 4) can be applied afterwards. All individuals are evaluated (step 5) by assigning them a fitness value based on the objective value and possible constraint violations. Thus a high fitness score means a good accommodation to the problem and corresponds with a small objective value. After evaluating each design the archive, which keeps good solutions, will be updated for the next selection step (step 6). The generation counter is increased and the iteration continues until a stopping criterion is fulfilled (step 7).

Figure 3.10: General Flowchart of Population Based Optimization Algorithms



The efficiency of population based methods can be significantly improved by choosing a suitable start population. If the optimization is performed after an initial sensitivity analysis, it is strongly recommended to use the best designs of the sensitivity analysis as start population. In order to keep the global character of the optimization it is useful to select designs covering a certain range of the design space.

The usage of population based algorithms is recommended in a wide range of applications. They are often the last resort for solving mathematically ill conditioned problems since their performance may be still good in case of a high number of design variables or a high amount of failed designs. Nevertheless, the convergence behavior may be slower compared to other optimization methods. The use of population based algorithms is recommended in all cases where gradient based optimization or response surface approximation fails, in case of a high number of variables or constraints, in case of discrete or binary design variables, in case of discrete responses or if the user is unaware about the optimization problem.

Release 2021 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Evolutionary Algorithms

Evolutionary algorithms (EA) are stochastic search methods that mimic processes of natural biological evolution. Many EA variants have been implemented over the past decades, based on the three main classes: genetic algorithms (GA) (Holland 1975 (p. 89)), evolution strategies (ES) (Rechenberg 1964 (p. 90)) and evolutionary programming (EP) (Fogel et al. 1966 (p. 89)). These algorithms have been originally developed to solve optimization problems where no gradient information is available, like binary or discrete search spaces, although they can also be applied to problems with continuous variables. Within optiSLang there is a flexible implementation of genetic algorithms and evolutionary strategies available which allows to scale between pure (strong) GA and pure (strong) ES.

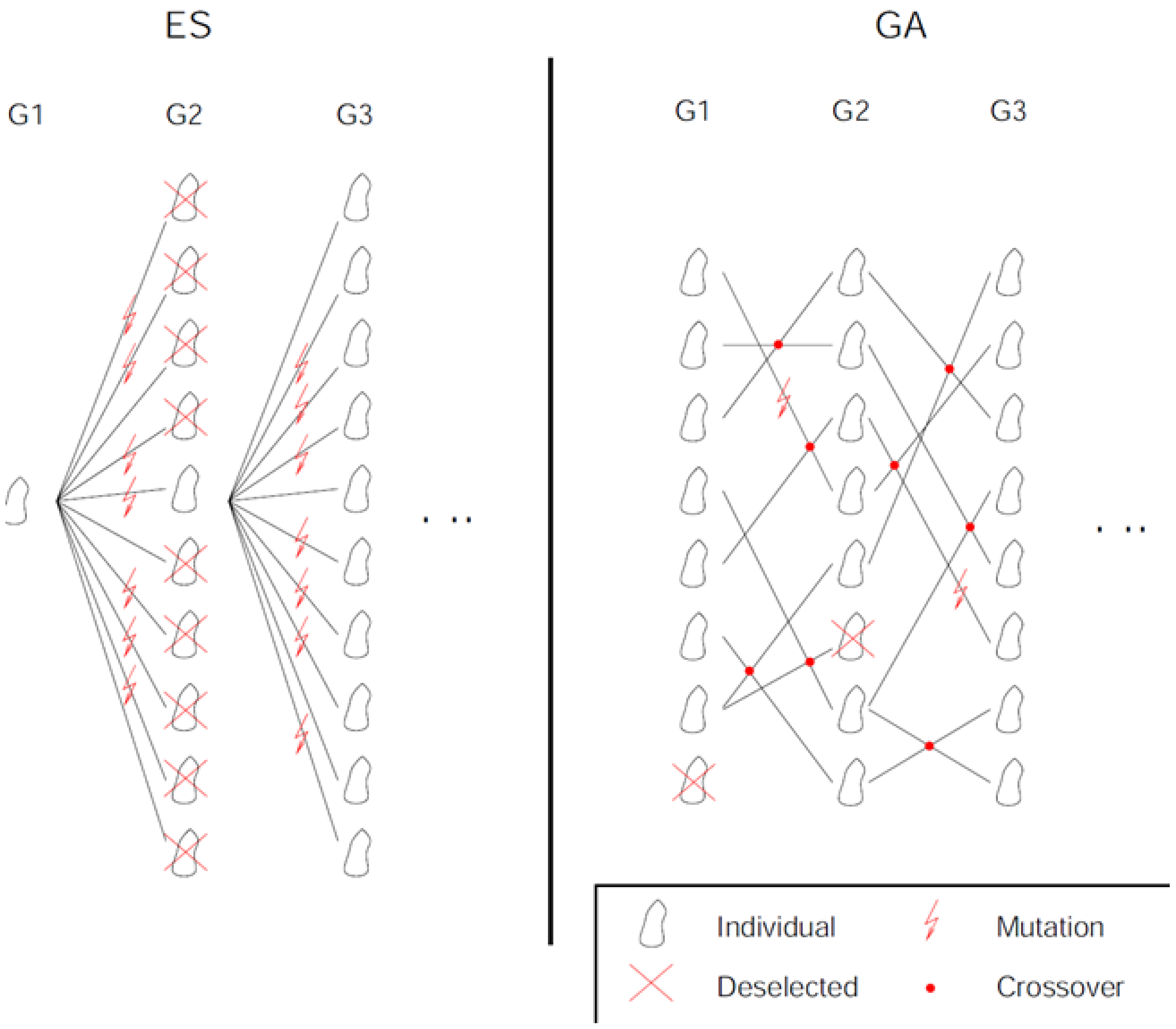
After the random or manual initialization of a population containing μ individid- uals the actual iteration loop starts, where every loop represents a generation g. Selection determines λ individuals for reproduction based on its fitness or ranking. Different stochastic selection operators are provided in order to adjust the selection pressure. Parent selection sets the direction of the stochastic search process. Variation is introduced by applying crossover and mutation operators to the selected individuals. These new individuals are called newborns or offspring. Finally the resulting offspring individuals are evaluated and a new population of μ individuals is formed using a replacement scheme.

The main difference between the two main variants, genetic algorithms (GA) and evolution strategies (ES), is the way variation is introduced to the population. Recombination of genes using crossover operators represents the main variation within genetic algorithms, while for evolution strategies (adaptive) mutation introduces variation to the population (see Figure 3.11: Evolution Strategies Versus Genetic Algorithms (p. 38)).

The **selection** of individuals for reproduction is based on random selection methods and requires an assignment of fitness values. A rank-based fitness assignment is used to overcome the scaling problem of the proportional fitness assignment. Due to the random selection and the fitness ranking, the probability of selecting a design with low fitness is low. However, this is not impossible, so also a weak design could be used for the generation of offspring.

The **crossover** operator is a method of recombination where two parent individuals produce two offspring by sharing information between chromosomes. The intention is to obtain individuals with better characteristics (exploitation) and to maintain the diversity of the population (exploration). Crossover is regarded as the main search operator in genetic algorithms.

Figure 3.11: Evolution Strategies Versus Genetic Algorithms



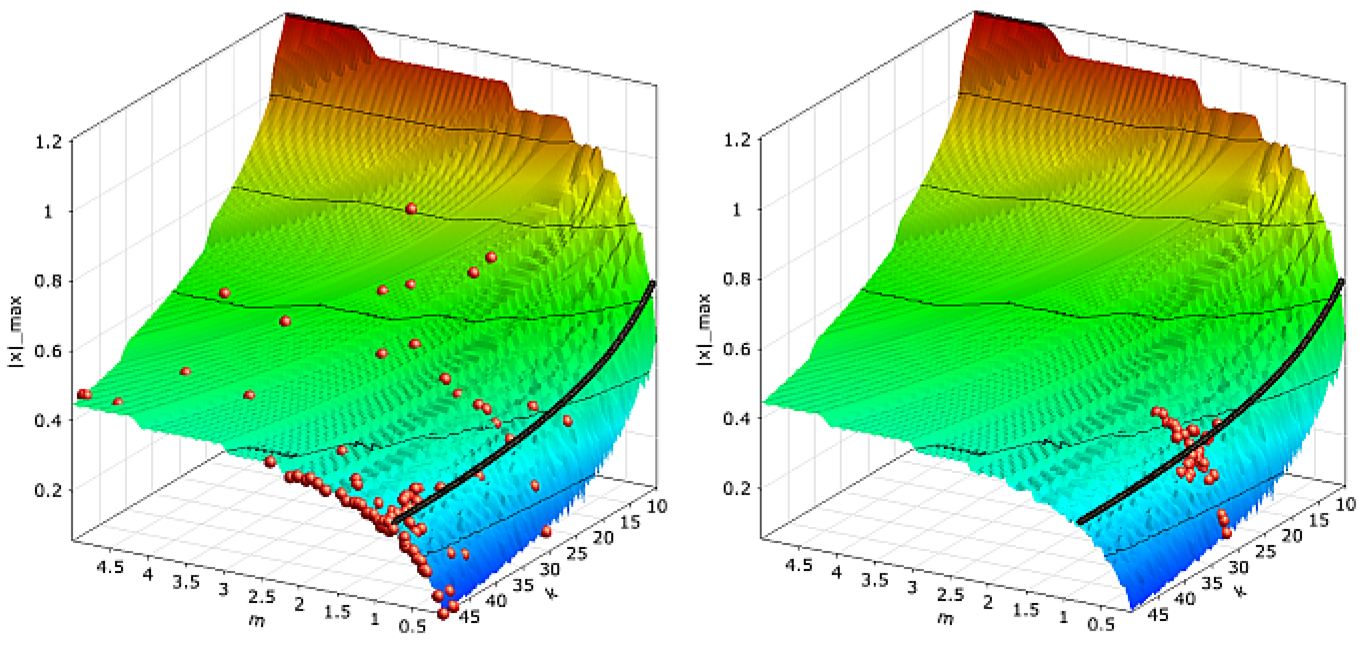
Mutation introduces random variation to the genes of the offspring chromosome. Each gene is selected for mutation with a specified probability or mutation rate respectively. The real-valued mutation is based on a normal distribution function for each gene with the value of the gene as its mean value. Mutation is the main search operator for evolutionary strategies (ES) but can also be applied after recombination (see Figure 3.10: General Flowchart of Population Based Optimization Algorithms (p. 36)). The mutation rate and the standard deviation defined in relation to the variable range are the control parameters of the mutation which can stay constant or get modified during the run of the algorithm. optiSLang provides constant and adaptive mutation procedures. Further details can be found in (Bäck 1996 (p. 89)) and (Riedel et al. 2005 (p. 90)).

The final operator, the archive update scheme, specifies how the population of the next generation is formed out of individuals of the current population (μ) and the generated offspring individuals (λ). The μ best individuals for the next generation are selected from a selection pool. Different strategies of constituting the selection pool are available by adjusting the size of the archive (α). The (μ, λ)-strategy with empty archive leads to a complete replacement of the population at every generation step with a maximum life span of an individual of only one generation. This method is used for classic genetic algorithms. The ($\mu + \lambda$)-strategy prevents archive individuals from being replaced by offspring with worse fitness. This strategy is recommended for use with evolution strategies. The ($\alpha(\mu) + \lambda$)-strategy allows the adjustment of the archive update scheme between the two extreme cases.

optiSLang provides a parameter-free constraint handling method taking into account all individuals of the population, which are compared regarding their objective values and constraint violations. The infeasible solutions are ranked according their constraint violations and their fitness is modified by adding the fitness of the worst feasible solution to the rank value. This procedure does not require additional penalty parameters. Furthermore it can handle multiple constraints without scaling of constraint values. Even if all individuals are violating constraints, the method can guide the search towards a feasible region of the constraint space.

In optiSLang a predefined global and local search is available for the evolutionary algorithms. The global search starts with a random or manually selected start population and is suitable to detect new possible solutions in the whole design space. The local search tries to improve a single design without discovering new regions. In Figure 3.12: Convergence of the Evolutionary Algorithm (p. 39), both search strategies are compared.

Figure 3.12: Convergence of the Evolutionary Algorithm



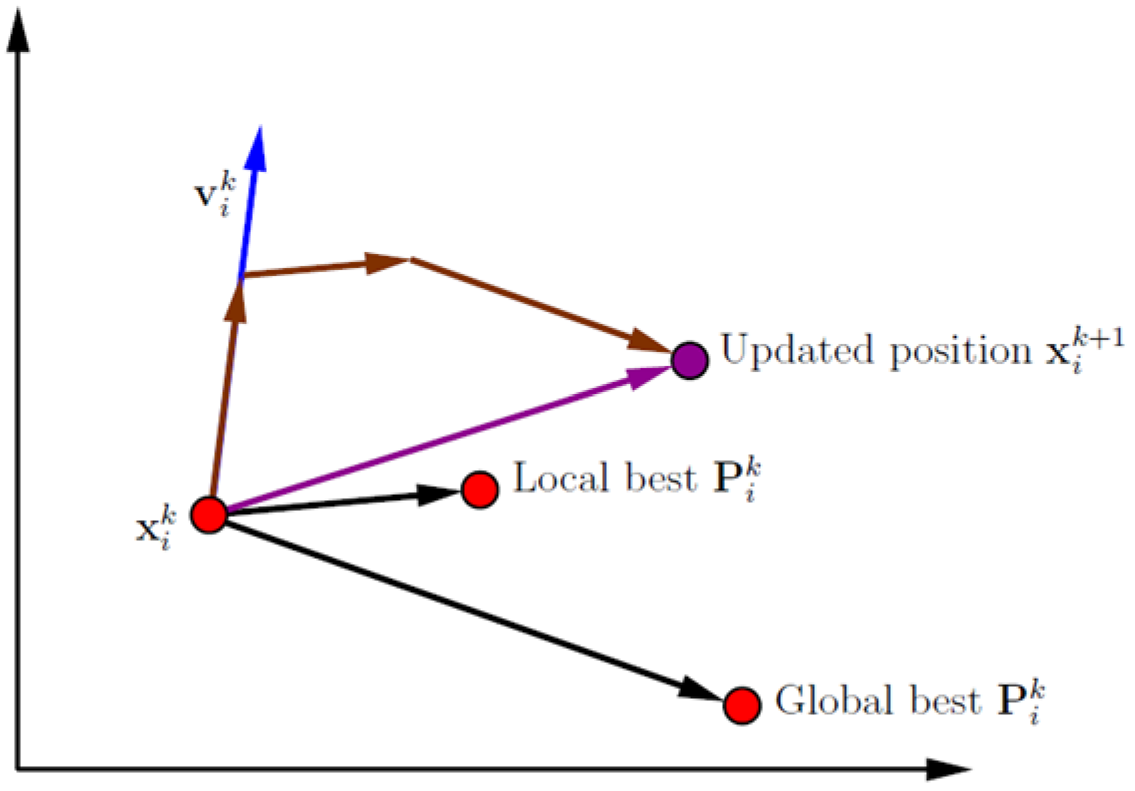
Convergence of the evolutionary algorithm with global search (left) and local search (right) for the damped oscillator with noisy objective function.

The figure indicates, that the global search, which starts with a random start population, generates designs in the whole design space while the designs generated with the local search method stay close to the start design. Due to the applied random search strategies in evolutionary algorithms the optimizer may not converge very close to the global optimum always. For example, the optimum found for the damped oscillator by using the global search (m = 0.81 kg, k = 50.0 N/m) is not as good as the results of the ARSM optimizer. For this reason it is often useful to use the evolutionary algorithms to detect the region of the global optimum and do further improvements with other optimization methods.

Particle Swarm Optimization

Particle Swarm Optimization (PSO) is a nature-inspired optimization method developed by (Kennedy and Eberhart 1995 (p. 89)) that imitates the social behavior of a swarm. Information about preferred positions will be transmitted to other individuals of the swarm such that they move into directions of previous optimal positions. In nature an individual can be represented by a bird, bee or fish. An overview of modifications and improvements of this class of algorithms can be found e.g. in (Engelbrecht 2005 (p. 89)) and (Poli et al. 2007 (p. 90)).

Figure 3.13: Particle Swarm Optimization


$\mathbf{v}_i^k$
Updated position $\mathbf{x}_i^{k+1}$
Local best $\mathbf{P}_i^k$
$\mathbf{x}_i^k$
Global best $\mathbf{P}_i^k$

Update of a particle position x_i^k in the Particle Swarm Optimization using a combination of the old velocity v_i^k, the direction to the local best position P_i^k and the direction to the global best position P_g^k.

PSO starts with initializing a population of size μ where each individual represents a possible solution of the optimization problem. Swarm intelligence is influenced by two main components representing the personal and global behavior. This means that each individual remembers its personal best solution and will also be influenced by the best solution of the swarm. Each individual will change its position $\mathbf{x_i}$ into the direction of its personal best found position $\mathbf{P_i}$ and the global best found position $\mathbf{P_g}$.

$$\begin{aligned} v_i^{k+1} &= w v_i^k + c_p R_1 \times (P_i^k - x_i^k) + c_g R_2 \times (P_g^k - x_i^k) \\ x_i^{k+1} &= x_i^k + v_i^{k+1} \end{aligned} \tag{3.6}$$

There are three important parameters that influence the speed and spread of the swarm. The inertia weight *w* is a scaling factor for the velocity of the previous iteration step *k*. The personal acceleration coefficient c_p is a scaling factor for the second term, which is also called cognitive component. The third term, called social component, is scaled by the global acceleration coefficient c_g . $\mathbf{R}_1$ and $\mathbf{R}_2$ are two vectors of random numbers uniformly chosen from [0, 1].

The choice of suitable coefficients c_p , c_g and w is very important for the convergence behavior of the PSO. optiSLang provides two predefined search strategies with default values for each coefficient. A local search strategy is recommended if the user has preliminary information about the optimization space and has already found a pre-optimized design. In this case the swarm movement is slower and less intensive throughout all generations. In a predefined global search strategy the swarm movement is very intensive at the beginning and will be damped throughout the optimization process by decreasing the weights of the previous velocity and the cognitive component and increasing the weight of the social component. In that way a global exploration is reached in the first iterations and exploitation is possible in last iterations.

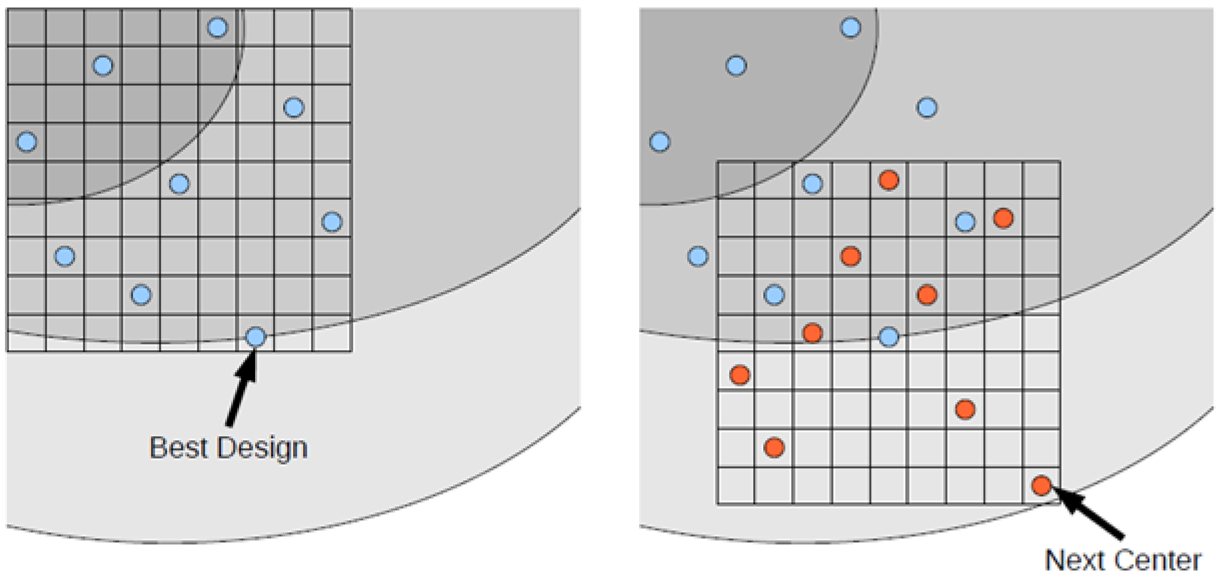
To allow a more sophisticated search, optiSLang enables the same mutation methods that are available for evolutionary algorithms. For executing a classical PSO search one has to deselect the mutation operator. The global best solution is recorded in the archive and is updated at the end of each iteration step. In the presence of constraint conditions the global best solution is found by taking the feasible solution with minimal objective value. If there are only infeasible solutions in the first population the global best will taken as the individual with least constraint violations.

The PSO works similarly to the evolutionary algorithms for optimization problems with a large amount of failed designs. It is less efficient compared to evolutionary algorithms if discrete design variables and many constraint conditions are investigated. For optimization tasks with continuous design variables the PSO converges generally very close to the global optimum. Similar to evolutionary algorithms, the choice of a qualified start population e.g. from sensitivity analysis may significantly improve the efficiency of the algorithm.

Stochastic Design Improvement

Stochastic Design Improvement (SDI) is a local, single-objective optimization procedure that improves a proposed design by using a simple stochastic approach without having extensive knowledge about potential relationships in design space. Based on an initial start design, a start population of size μ is generated by an uniformly distributed Latin Hypercube Sampling within a given range. In each iteration step, the best design is taken as new center point for the sampling of the next iteration as shown in Figure 3.14: Stochastic Design Improvement (p. 42). The ranges of the sampling scheme are adapted in each iteration step. Depending on the optimization problem the whole population might move into a better region and achieve an improvement in each step. All individuals of one iteration will be compared concerning objective and constraints by using the parameter-free constraint handling method applied in the evolutionary algorithms. The algorithm converges if either a maximum number of iterations was reached, if a specified improvement with respect to the initial design was reached (gained improvement) or if a specified deterioration of the performance between two iteration steps was observed.

Figure 3.14: Stochastic Design Improvement



Update scheme of the Stochastic Design Improvement approach by generating a random sampling around the best design of a previous iteration.

Due to its pure stochastic approach the SDI is very robust and works for a high amount of failed designs, discrete and continuous parameters, and a high number of design variables and constraint conditions. Nevertheless, it was designed to improve an initial design but not to find global optimal solutions. For this reason its efficiency may be significantly lower compared to evolutionary algorithms and Particle Swarm Optimization.

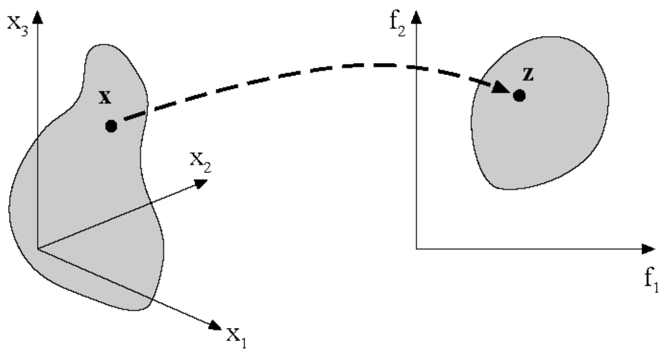
3.2. Multi-Objective Optimization

Most real-world optimization problems include more than one objective and a large number of design variables. This leads to the general formulation of the multi-objective optimization problem

$$\begin{aligned} \text{minimize:}\, & f_m(\mathbf{x}), \quad m=1,2,\ldots,M \\ \text{subject to:}\, & g_j(\mathbf{x}) \geq 0, \quad j=1,2,\ldots,J \\ & h_k(\mathbf{x})=0, \quad k=1,2,\ldots,K \\ & x_i^{(L)} \leq x_i \leq x_i^{(U)}, \; i=1,2,\ldots,n \end{aligned} \tag{3.7}$$

with $\mathbf{x}=(x_1,x_2,\ldots,x_n)^T$ as the vector of design variables and the constraints of inequality $g_j(\mathbf{x})$ and equality $h_k(\mathbf{x})$. All solutions which comply with both constraints and variable bounds form the feasible n-dimensional design space. Each solution $\mathbf{x}$ is assigned to a vector $\mathbf{f}(\mathbf{x})=\mathbf{z}=(z_1,z_2,\ldots,z_M)^T$ describing one point of the M -dimensional objective space. This is the major difference to the single-objective optimization problem where there is only one objective. The mapping between design space and objective space is illustrated in Figure 3.15: Design Space (Left) and Objective Space (Right) (p. 43).

Figure 3.15: Design Space (Left) and Objective Space (Right)



Several procedures have been developed to solve a multi-objective optimization problem (Branke et al. 2008 (p. 89)). They are classified in non-interactive and interactive methods. In non-interactive methods the multi-objective optimization problem is transferred into a single-objective problem. This can be done e.g. by choosing a preferred objective function and introducing the remaining objectives as constraints

$$\begin{aligned} \tilde{f}(x) &= f_i(x) \\ f_{j \neq i}(x) &\leq f_j^{\text{limit}}. \end{aligned} \tag{3.8}$$

Another possibility is to combine all objectives in a single function by using individual weights w_i

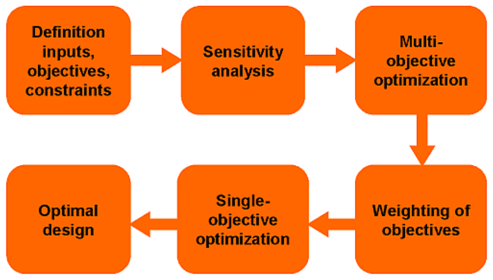
$$\tilde{f}(x) = \sum_{i=1}^{M} w_i f_i(x). \tag{3.9}$$

Both methods require good knowledge about the optimization potential with respect to each objective function and a clear idea about the importance of the different objectives with respect to each other.

In early stages of the design process such information are often rare and a transformation of the multi-objective task into a single objective task is not possible a priori. In such cases interactive methods are more promising. In optiSLang Pareto optimization as one of the most popular interactive methods is available.

In Figure 3.16: Recommended Flowchart for Multi-Objective Optimization (p. 44), the recommended flow for a multi-objective optimization procedure is shown: initially the inputs, possible objectives and constraint functions are defined. Sensitivity analysis is used to detect unimportant parameters and to check the objective functions with respect to possible conflicts. Afterwards, a multi-objective optimization is performed to determine the optimization potential within the conflicting objectives and to derive suitable weighting factors for a following single-objective optimization. Finally this single-objective optimization determines an optimal design.

Figure 3.16: Recommended Flowchart for Multi-Objective Optimization


Definition inputs, objectives, constraints
Sensitivity analysis
Multi-objective optimization
Weighting of objectives
Single-objective optimization
Optimal design

3.2.1. Pareto Optimization

Pareto Optimality

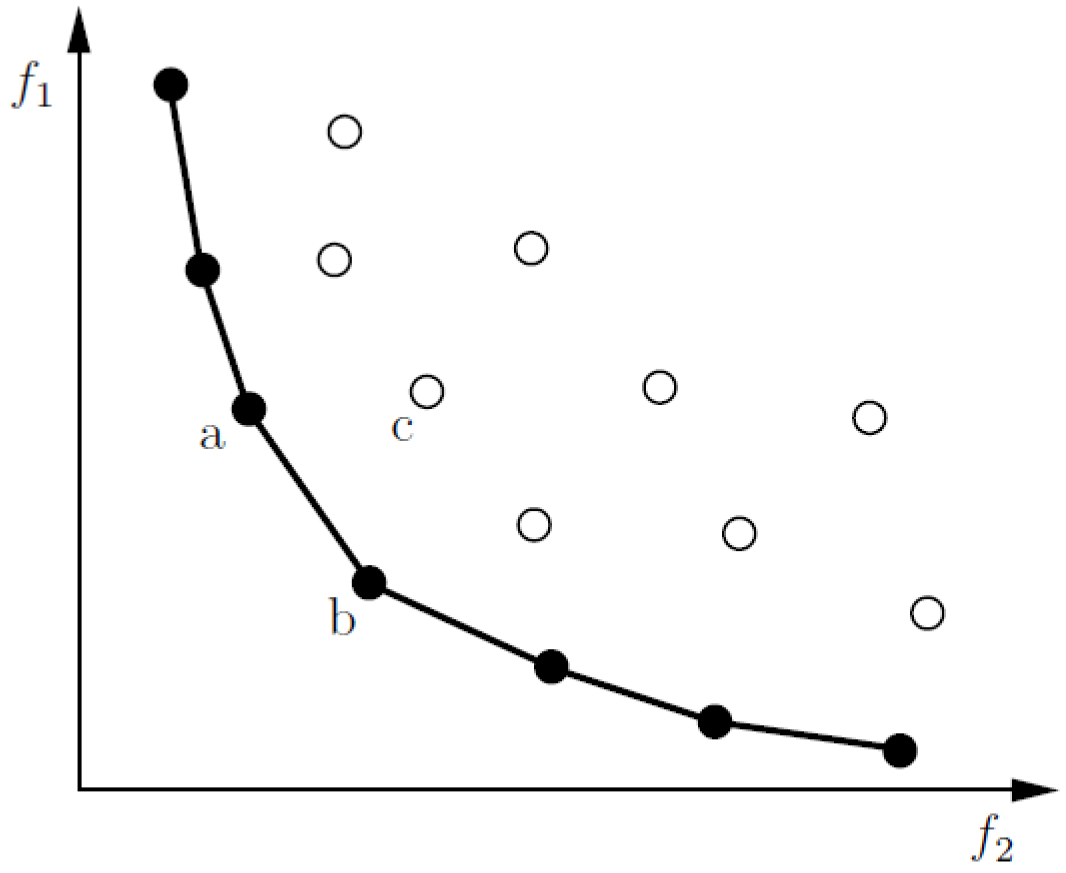
Solving the multi-objective optimization problem requires a criterion which regards the distinct objectives. One possible criterion is the Pareto dominance which is formulated for two decision vectors **a** and **b** in the design space X as follows:

- Solution **a** $\in X$ dominates solution **b** $\in X$ if it is feasible and better or equal in all objectives and better in at least one objective.
- Solution **a** $\in X$ is indifferent to solution **b** $\in X$ if none of both solutions dominates the respective other.

Using the dominance criterion allows for creating a partial order between different decision vectors as illustrated in Figure 3.17: Pareto (p. 45). A solution is Pareto optimal if there is no decision vector that would improve one objective without causing a deterioration in at least one other objective. In other words a solution is Pareto optimal if it is not dominated by any other solution. The term Pareto is named after Vilfredo Pareto, an Italian economist who used the concept in his studies of economic efficiency and income distribution.

If all objectives are equally important and no preferences are made a priori, the dominance of a solution is the only way to determine if it is better than others. As a result the non-dominated subset out of the feasible set of solutions constitutes the Pareto set. The corresponding points in the objective space are called Pareto frontier.

Figure 3.17: Pareto



Pareto dominating (filled circles) and dominated designs (unfilled circles) including the resulting Pareto frontier of two conflicting objectives. (*a* and *b* dominate *c* but are indifferent to each other)

The following requirements concerning the multi-objective optimization can be formulated:

- Find a set of solutions close to the Pareto optimal solutions (convergence).
- Find solutions which are diverse enough to represent the whole Pareto frontier (diversity).

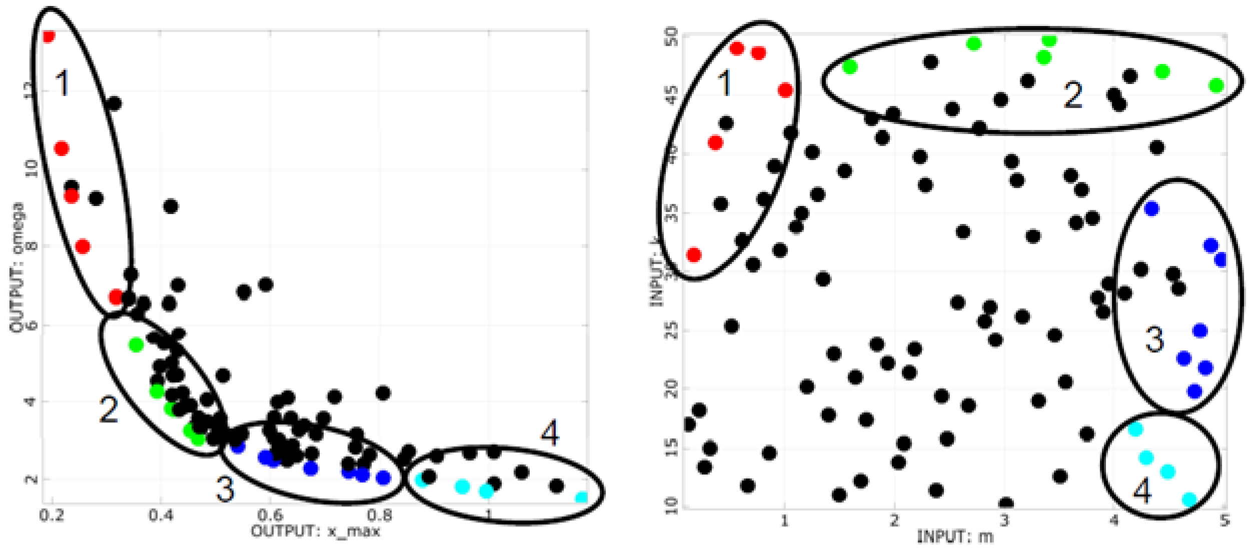
Although the optimization produces a set of Pareto optimal solutions, the user is interested in getting only one solution. Additional preferences have to be introduced, which needs comprehensive knowledge about the problem, to select a solution that meets the requirements.

Analysis of Conflicting Objectives

In order to obtain different trade-off solutions there must be conflicting objectives. The results of a preceding sensitivity analysis should be used to check the objective functions concerning possible conflicts.

In case of the oscillator example the damped eigen-frequency is used as second objective function which should be minimized. The anthill plots of the samples used for the sensitivity analysis and design exploration are shown in Figure 3.18: Conflicting Objectives (p. 46). The figure indicates that the minimization of the maximum amplitude is in conflict with the minimization of the damped eigen-frequency. In the anthill plot all Pareto dominant designs can be identified. Further information can be gained by subdividing the Pareto optimal designs in the objective space in clusters which are investigated with respect to their region in the design space. The figure indicates e.g. that the designs of cluster 2 are highly concentrated in the objective space but widely distributed in the design space. Furthermore, all Pareto optimal designs are close to the design space boundaries.

Figure 3.18: Conflicting Objectives



Conflicting objectives of the damped oscillator (maximum amplitude vs. eigen-frequency) including cluster analysis of the Pareto optimal designs in the objective space (left) and in the design space (right).

Multi-Objective Population-Based Methods

The application of evolutionary algorithms for solving multi-objective optimization problems has been established over the past two decades. The parallel search for a set of Pareto optimal solutions is the major advantage of this method. The optiSLang implementation is based on the *Strength Pareto Evolutionary Algorithm 2* (SPEA2) (Zitzler et al. 2001 (p. 90)) and is characterized by the following principles:

- Elitism is applied by using an archive of non-dominated individuals.
- Fitness assignment is based on the dominance criterion.
- Preservation of population diversity is realized by density estimation.

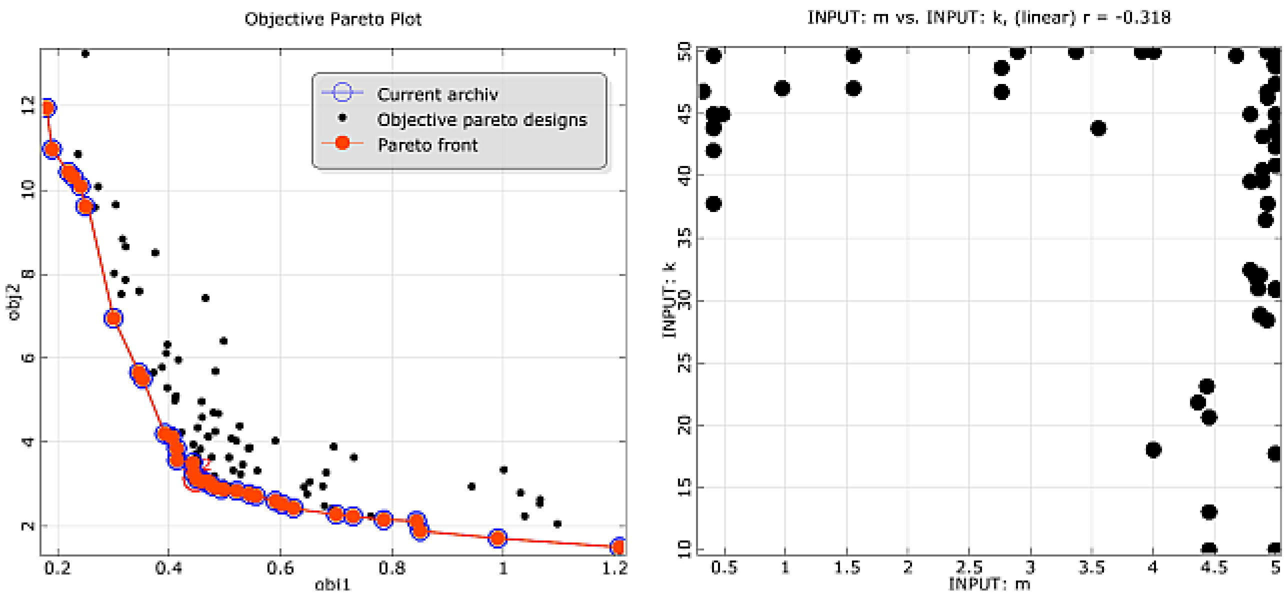
The algorithm starts with the random or manual initialization of the first population of size μ followed by the evaluation of all individuals. For the first generation λ (number of parents) individuals are selected out of the population for reproduction. After the evaluation of all newborns both archive individuals and offspring are assigned new fitness values. The α best solutions form the archive of the next generation. The optimization stops if the maximum number of generations has been reached or the archive stagnated.

Release 2021 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The Particle Swarm Optimization algorithm can also be applied to multi-objective optimization problems. As mentioned for single-objective PSO optiSLang selects the fittest design as global leader. For the next iteration step the swarm will move in this direction. In multi-objective optimization the fitness evaluation is different from the single-objective approach.

In contrast to the single-objective nature inspired optimization methods, fitness assignment of the multi-objective EA and PSO always regards the whole population, because the criterion is based on the comparison of individuals. The fitness assignment method is a dominance-based ranking which takes into account by how many individuals a solution is dominated and how many individuals a solution dominates. To preserve the diversity of the Pareto frontier an additional criterion is introduced to the fitness assignment procedure. The distance of an individual to its *k*-th nearest neighbor serves as an estimator of density. The intention is to prefer solutions in less crowded regions of the objective space such as boundary solutions.

Figure 3.19: Pareto Frontier



Pareto frontier for the damped oscillator (left) and Pareto optimal designs plotted in the design space (right) obtained by evolutionary algorithms.

In Figure 3.19: Pareto Frontier (p. 47), the resulting Pareto frontier of the multi-objective evolutionary algorithm is shown for the damped oscillator example. The Pareto frontier shows a good agreement with the Pareto optimal designs from Figure 3.18: Conflicting Objectives (p. 46). The Pareto frontier can now be used to judge about the importance of the different optimization goals and to choose a suitable optimal design. The efficiency of the multi-objective optimization methods can be dramatically improved especially for high-dimensional optimization problems by choosing a suitable start population instead of a pure random generation. Selected designs of a sensitivity analysis or previous optimization runs shall be used for this purpose.

Release 2021 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Chapter 4: Robust Design

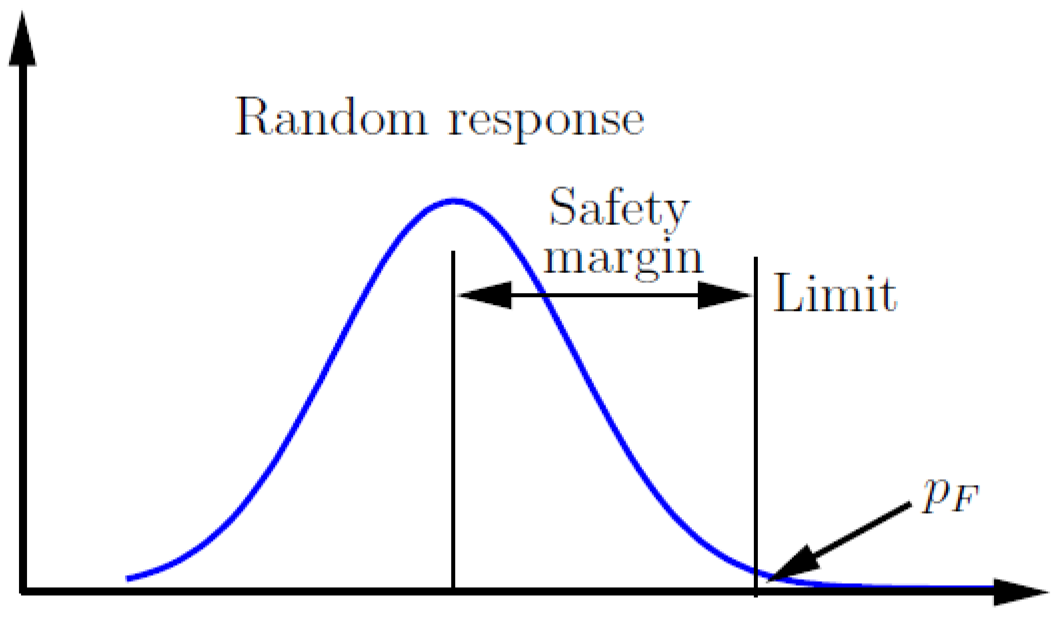
Due to target-oriented, automatic optimization of virtual products new design possibilities are explored. However, highly optimized designs lead to high imperfection sensitivities and tend to loose robustness. Often the deterministic optimum is pushed to the boundaries of the feasible design space. As a result the optimized design, which was found by assuming deterministic model properties, may not be realizable in a production process. For this reason it is necessary to investigate, how the optimized design is affected by scattering model input variables, which could be e.g. geometry and material parameters, boundary conditions and loads. The scattering inputs can be modeled within optiSLang by means of scalar random variables having a certain dependence between each other. Random variables have the advantage compared to other uncertainty models, that efficient methods of the well developed probability theory can be applied.

A robust design may be characterized intuitively in that way, that its performance is largely unaffected by random perturbations of the model inputs. A possible measure is the variance indicator, where the relative variations of the critical model responses are compared to the relative variation of the input variables. If certain model responses are limited with respect to an undesired performance, the safety margin can be quantified as the interval between the mean value of the model response and the limit. This is shown in Figure 4.1: Safety Margin (p. 50). The safety margin can be formulated in terms of the standard deviation of the model response. In the variance-based robustness analysis a specific safety margin $\alpha\sigma_Y$, which has to be defined by the designer, has to be proven for all critical responses

$$\left| y\mathrm{limit} - \overline{Y} \right| \le \alpha\sigma_Y. \tag{4.1}$$

Alternatively the probability that a certain limit is exceeded can be quantified and proven to be less than an acceptable value. This probability indicator can be eval- uated by the probability-based robustness analysis, often called reliability analysis.

Figure 4.1: Safety Margin



Random model response with given limit value and corresponding safety margin and failure probability p_F.

4.1. Definition of Uncertainties

See the following sections:

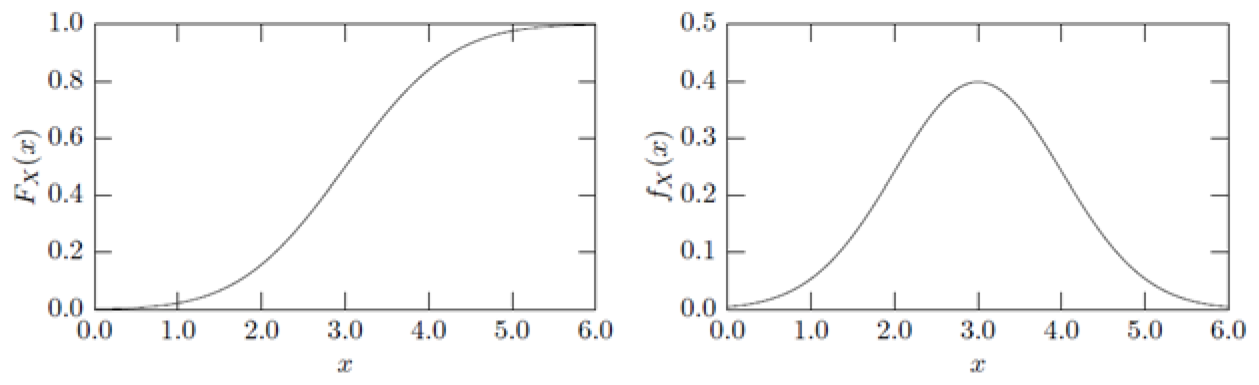
4.1.1. Scalar Random Variables

Random variables are scalar quantities which may be represented by a set of discrete values, each having a specific probability, or by continuous variables defined in a given range. A continuous random variable *X* is described by the cumulative distribution function

$$F_X(x)=P[X<x],\ \lim_{x\to-\infty}F_X(x)=0,\ \lim_{x\to-\infty}F_X(x)=1, \tag{4.2}$$

where *P* [*X* < *x*] is the probability that *X* is lower than a given value *x*.

Figure 4.2: Cumulative Distribution Function and Probably Density Function



Cumulative distribution function $F_X(x)$ and probability density function $f_X(x)$ of a continuous random variable *X*.

Equivalent to the cumulative distribution function often the probability density function, which is the derivative of the cumulative distribution function, is utilized

$$f_X(x)=\frac{\partial F_X(x)}{\partial x},\ F_X(x)=\int_{-\infty}^{x} f_X(\tau)d\tau,\ \int_{-\infty}^{\infty} f_X(\tau)d\tau=1. \tag{4.3}$$

In Figure 4.2: Cumulative Distribution Function and Probably Density Function (p. 51), the cumulative distribution function and the probability density function of a continuous random variable are shown exemplary.

Based on the density function, stochastic moments have been introduced, see for example (Bucher 2009). The first, absolute moment of a random variable *X* is its mean value

$$\overline{X}=E[X]=\int_{-\infty}^{\infty} x f_X(x)dx, \tag{4.4}$$

where *E*[·] is the expected value. The second, central moment of a random variable is its variance

$$\sigma_X^2=E[(X-\overline{X})^2]=\int_{-\infty}^{\infty}(x-\overline{X})^2 f_X(x)dx. \tag{4.5}$$

Mean value and variance (or standard deviation $\sigma_X=\sqrt{\sigma_X^2}$) are typically used to describe the variation of a random variable. However, even the shape of the distribution function may have a significant influence on the results of a stochastic analysis. Specific distribution types have been proposed during the last centuries. The most famous is the normal (Gaussian) distribution. Other common distribution types are e.g. uniform, log-normal, exponential, Gumbel, and Weibull distributions. In optiSLang a large number of well-known distribution types is available. The probability density functions and properties are given in Distribution Types (p. 80). Further details on distribution types of random variables can be found for example in (Montgomery and Runger 2003 (p. 90)) and (Bucher 2009 (p. 89)).

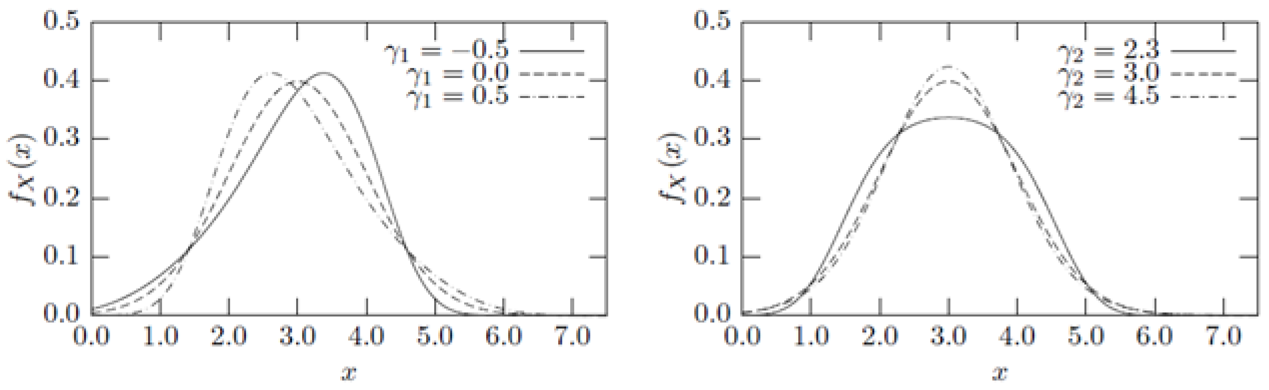
In order to perform a robustness analysis, optiSLang requires the definition of distribution types including mean value and standard deviation for all random input variables. Since the stochastic properties of the inputs have a significant influence on the results of a robustness analysis, the user is requested to consider as much information as possible for this purpose. If measurements should be considered, the statistic post processing of optiSLang can be used to estimate the stochastic moments and to find a suitable distribution type.

Additionally to mean value and standard deviation often higher order moments of random model responses and even of random inputs are of interest. Skewness $\gamma 1$ and kurtosis $\gamma 2$ are the standardized third and fourth moments of a random variable

$$\gamma 1=\frac{E[(X-\overline{X})^3]}{\sigma_X^3},\ \gamma 2=\frac{E[(X-\overline{X})^4]}{\sigma_X^4}. \tag{4.6}$$

They provide a description of the shape of a random variable density function in addition to mean value and standard deviation. For example, for symmetric density functions the skewness is always zero and for the normal distribution the kurtosis is equal to three. In Figure 4.3: Density Functions with Corresponding Skewness and Kurtosis Values (p. 52), several density functions are shown together with corresponding skewness and kurtosis values.

Figure 4.3: Density Functions with Corresponding Skewness and Kurtosis Values



Probability density function $f_X(x)$ of a continuous random variable X for different values of skewness $\gamma 1$ and kurtosis $\gamma 2$.

4.1.2. Multivariate Distributions

It is typical for industrial applications to investigate the influence of a large number of random input variables. Modelling these input variables as independent is a simplification, but often the consideration of dependencies between inputs has a significant influence on the results obtained. For example, material properties like the Young's modulus and yield stress may not vary independently. In optiSLang dependent random variables are modeled with help of the coefficient of correlation

$$\rho(X_1,X_2)=\frac{E[(X_1-\bar{X}_1)(X_2-\bar{X}_2)]}{\sigma_{X_1}\sigma_{X_2}}. \tag{4.7}$$

In the following, the random input variables of an investigated model are col- lected in a random vector **X**. The joint probability density function is used to quantify the probability of a specific combination of values of inputs. In the case that all random variables X_i are independent from each other, then the joint probability density is the product of the single (marginal) density functions.

$$f_{\mathrm{X}}(\mathrm{x})=\prod_{i=1}^{n} f_{X_i}(x_i) \tag{4.8}$$

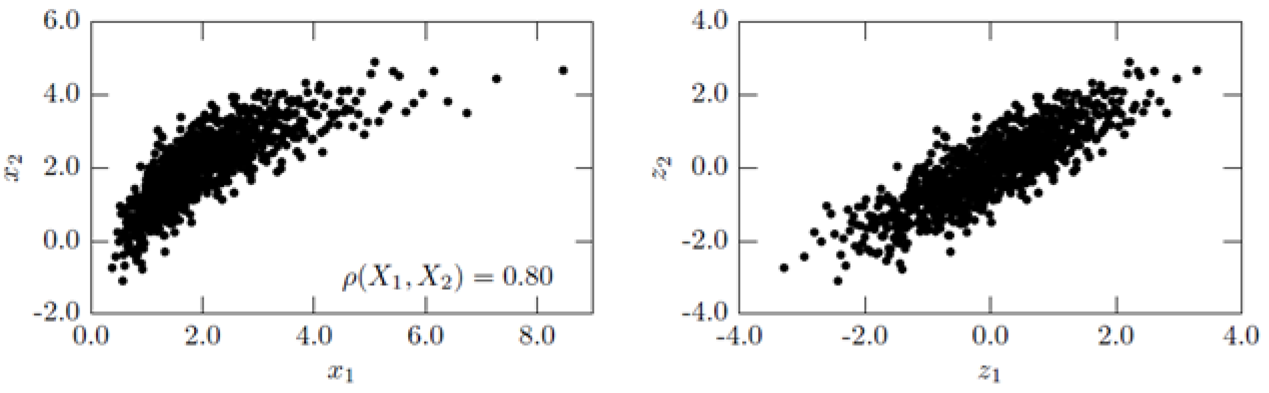
Note that independent variables are always uncorrelated, while the contrary is not always true. For normally distributed input variables **X** $_g$ the joint density function reads

$$f_{X_g}(x) = \frac{1}{\sqrt{(2\pi)^n |C_{X_g X_g}|}} \exp\left[-\frac{1}{2}(x-\bar{X}_g)^T C_{X_g X_g}^{-1}(x-\bar{X}_g) \right], \tag{4.9}$$

where $\overline{X}$ is the vector of mean values, C_{XX} is the covariance matrix and n is the number of random variables. The combination of arbitrary distribution types is not possible in closed form. In optiSLang, the combination of correlated variables with arbitrary distribution types is realized by using the Nataf model (Nataf 1962 (p. 90)).

First, the single original random variables are transformed to Gaussian ones. For the assumed joint normal probability density, the coefficients of correlation differ from the original values, they have to be found by iteration. It may happen in special cases that the iteration yields a not positive definite, hence invalid matrix of correlation coefficients. In (Bucher 2009 (p. 89)), further details about this procedure can be found.

Figure 4.4: Random Samples of a Two-Dimensional Random Vector



Samples of a log-normal – normal distributed random vector obtained by the Nataf model: in the original space (left) and in the normal space (right).

In Figure 4.4: Random Samples of a Two-Dimensional Random Vector (p. 53), random samples of a two-dimensional random vector with a normal and a log-normal distribution type are shown. The transformed samples in the normal space can be compared to the identical samples in the original space. This indicates, that although linear dependencies are assumed in the normal space, non- linear dependencies can be represented in the original space.

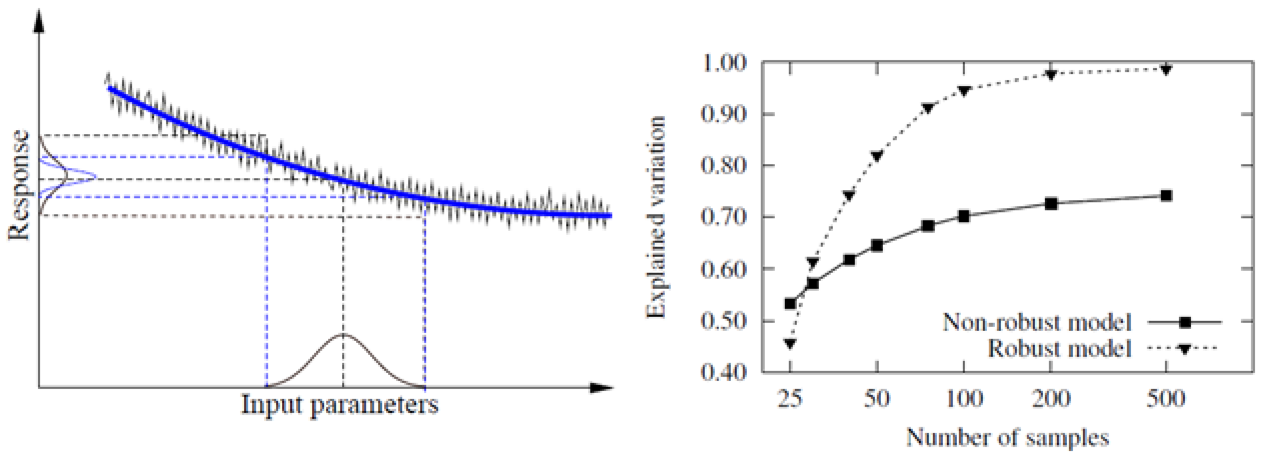
4.2. Variance-Based Robustness Analysis

In variance-based robustness analysis the variations of the critical model responses are investigated. In optiSLang random sampling methods are used to generate discrete samples of the joined probability density function of the given random variables. Based on these samples, which are evaluated by the solver similarly as in the sensitivity analysis, the statistical properties of the model responses as mean value, standard deviation, quantiles and higher order stochastic moments are estimated. In order to obtain a sufficient quality of these estimates, it is required, that the sampling scheme represents the marginal distributions of the single random variables as well as the defined correlations between each other with high accuracy. Some very basic stochastic methods to generate sample sets are variants of the Monte-Carlo method. The simplest version is the so-called plain Monte-Carlo method (PMC). With this methods the natural scatter can be modeled quite well, but the statistical uncertainty is fairly large if the sample size is small. Therefore optiSLang provides Latin Hypercube Sampling (LHS) with minimized correlation errors (Hungtington and Lyrintzis 1998 (p. 89)), where the marginal distributions and the predefined input correlations are represented with a small number of samples.

Based on the estimates of the mean value and the standard deviation the safety margin can be estimated by using Equation 4.1 (p. 49) for the responses where a performance limit is given. However, by using variance-based robustness analysis only safety margins up to two sigma can be proven with a small number of samples. For larger safety margins (e.g. six sigma) the true failure rate may be heavily vary for different distribution types of the output. Since the distribution of the output is not exactly known, an estimate of low failure probabilities by variance-based measures may be very inaccurate. Therefore, safety margins larger than three sigma should be proven by reliability analysis.

Additional to the variation of the response, the mean value could be an indicator regarding the robustness of a design. If the mean value estimated by the samples strongly differs from the deterministic reference value, the influence of the input uncertainties and potentially of solver inaccuracies is significant.

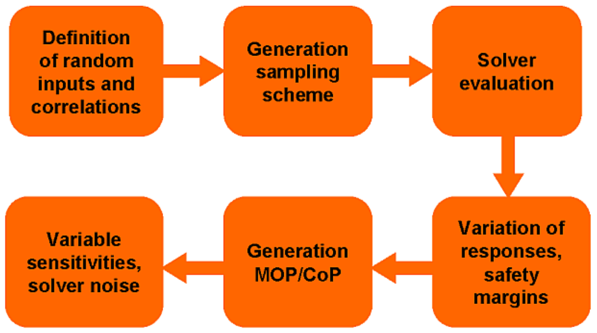
Figure 4.5: Noisy Model Outpot and Convergence Behavior



Representation of a noisy model output by a smoothing approximation (left) and convergence behavior of the explained variation in the case of a robust and non-robust model (right).

In order to quantify the sources of the response variation, in optiSLang variance-based sensitivity measures are estimated within the robustness analysis. For this purpose the Meta-model of Optimal Prognosis presented in Metamodel of Optimal Prognosis (p. 18) is applied. If the solver output contains unexplainable effects due to numerical accuracy problems, the MOP approximation will smooth these noise effects as shown in Figure 4.5: Noisy Model Outpot and Convergence Behavior (p. 54). If this is the case, the CoP value of the MOP can be used to estimate the noise variation as the shortcoming of the CoP to 100% explainability. However, the unexplained variation may not be caused by solver noise only, but also by a poor approximation quality. This problem should be analyzed by increasing the number of samples in the case of low explainability. If the CoP does not increase, then this is an indicator for unexplainable solver behavior.

Figure 4.6: Flowchart of Variance-Based Robustness Analysis Available in optiSLang


Definition of random inputs and correlations
Generation sampling scheme
Solver evaluation
Variation of responses, safety margins
Generation MOP/CoP
Variable sensitivities, solver noise

In Figure 4.6: Flowchart of Variance-Based Robustness Analysis Available in optiSLang (p. 55), the overall flow of the variance-based robustness analysis provided by optiSLang is illustrated. Based on the initial definition of random variables, optimized samples are generated and evaluated by the solver. The solver results are used to estimate the statistical properties and to perform a sensitivity analysis with help of the MOP approach. If the design does not fulfill the robustness requirements, the sensitivity indices help to identify these input variables, where a reduction of the uncertainty is most effective with respect to the variation of the model responses.

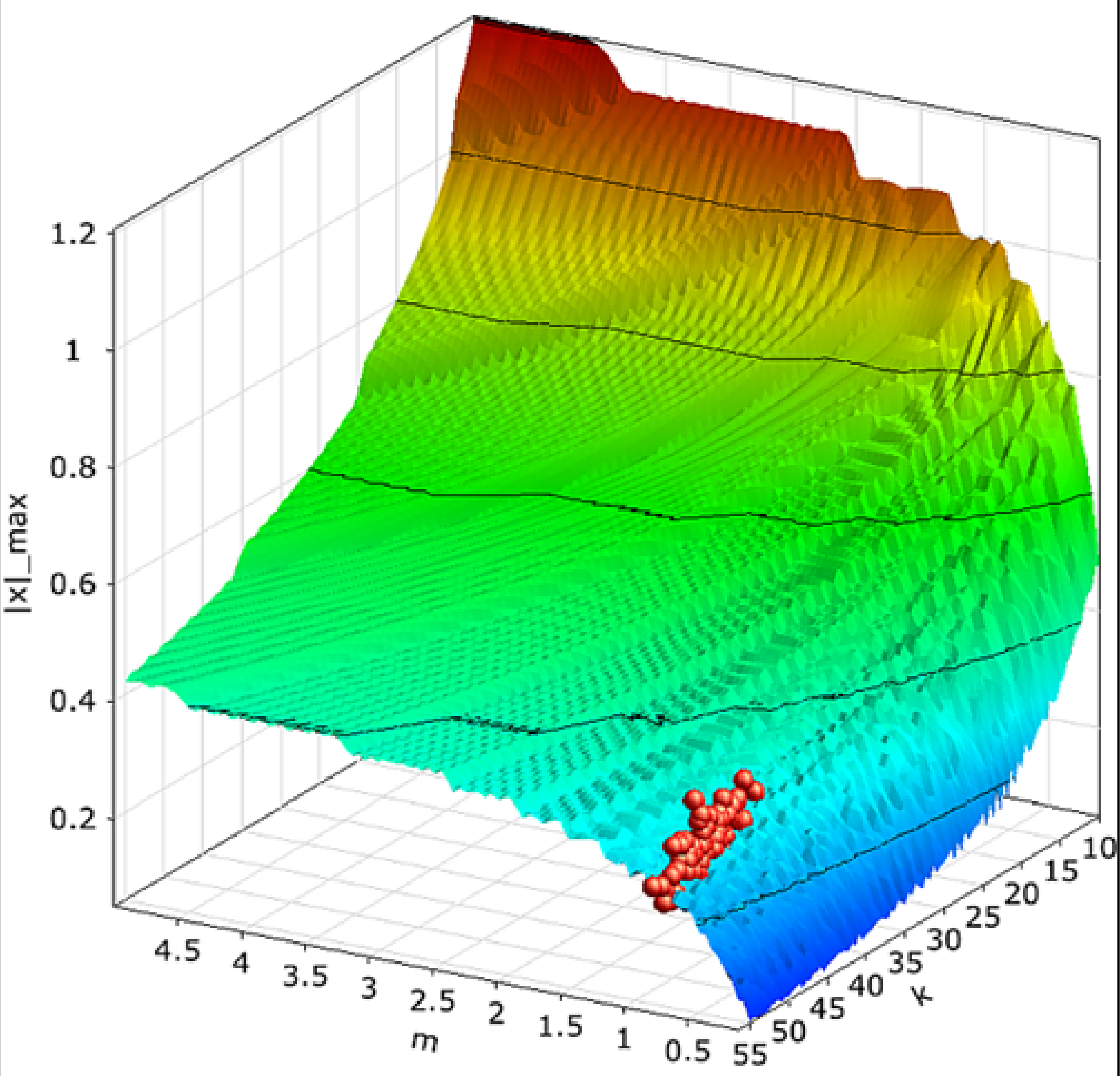
Exemplary, the variance-based robustness evaluation is performed at the deterministic optimum of the damped oscillator introduced in Single-Objective Optimization (p. 27). The mass *m*, the spring stiffness *k*, the damping ratio *D* and the initial kinetic energy E_{kin} are considered as independent random variables. Their mean values and coefficients of variation (CV) are taken as follows

$$\begin{aligned}\mu_m &= 0.78\,kg, \quad CV_m=2\%, \quad \mu_k=50.0N/m, \quad CV_k=5\%,\\ \mu_D&=0.02, \quad CV_D=10\%, \quad \mu_E=10.0Nm, \quad CV_E=10\%.\end{aligned} \tag{4.10}$$

100 Latin Hypercube samples, which are shown in Figure 4.7: Variance-Based Robustness Analysis of the Damped Oscillator (p. 56), are evaluated. As robustness requirement the safety margin of the mean eigen-frequency to a given limit ω_{limit} = 8.5 1/s shall be larger or equal 4.5 sigma, which is equivalent to a failure probability of $3.4 \cdot 10^{-6}$ for a normal distribution. Furthermore, the mean values and the relative variation of the model responses as well as their explained variations are statistically evaluated.

In Figure 4.8: Statistical Properties of the Oscillator Responses and Input Sensitivities Obtained with Variance-Based Robustness Analysis (p. 58) the statistical properties of the maximum amplitude and the damped eigen-frequency are visualized. The figure indicates, that the coefficient of variation of the maximum amplitude is larger (11%) as the largest CV of the input variables. Furthermore, the mean value is almost one sigma larger as the deterministic value ($|x|^{opt}_{\max}$ = 0.25 m). Both facts could be interpreted as indicators for a non-robust design in terms of the objective.

Figure 4.7: Variance-Based Robustness Analysis of the Damped Oscillator



100 Latin Hypercube samples used for the variance-based robustness analysis of the damped oscillator.

As result of the MOP-based sensitivity analysis the damping ratio has the largest contribution to the variation of the maximum amplitude. The Coefficient of Prognosis indicates an explained variation of 92% which is smaller as the CoP obtained with the optimization variables, which was 99%. Since the robustness analysis considers in this example a much smaller variation in the random inputs as the sensitivity analysis in the optimization variables, the nonlinearity in the model response should be reduced in the smaller space. The decrease of the CoP in that case indicates an increasing importance of solver noise which is caused by the coarse time discretization of the oscillator response. The statistical evaluation of the damped-eigen-frequency results in a CV of 2.7% which is in the range of the input variation. Furthermore, the CoP indicates an excellent explainability of 100% for this model response. Nevertheless, the require safety margin of 4.5 sigma is not reached. The mean value and the standard deviation given

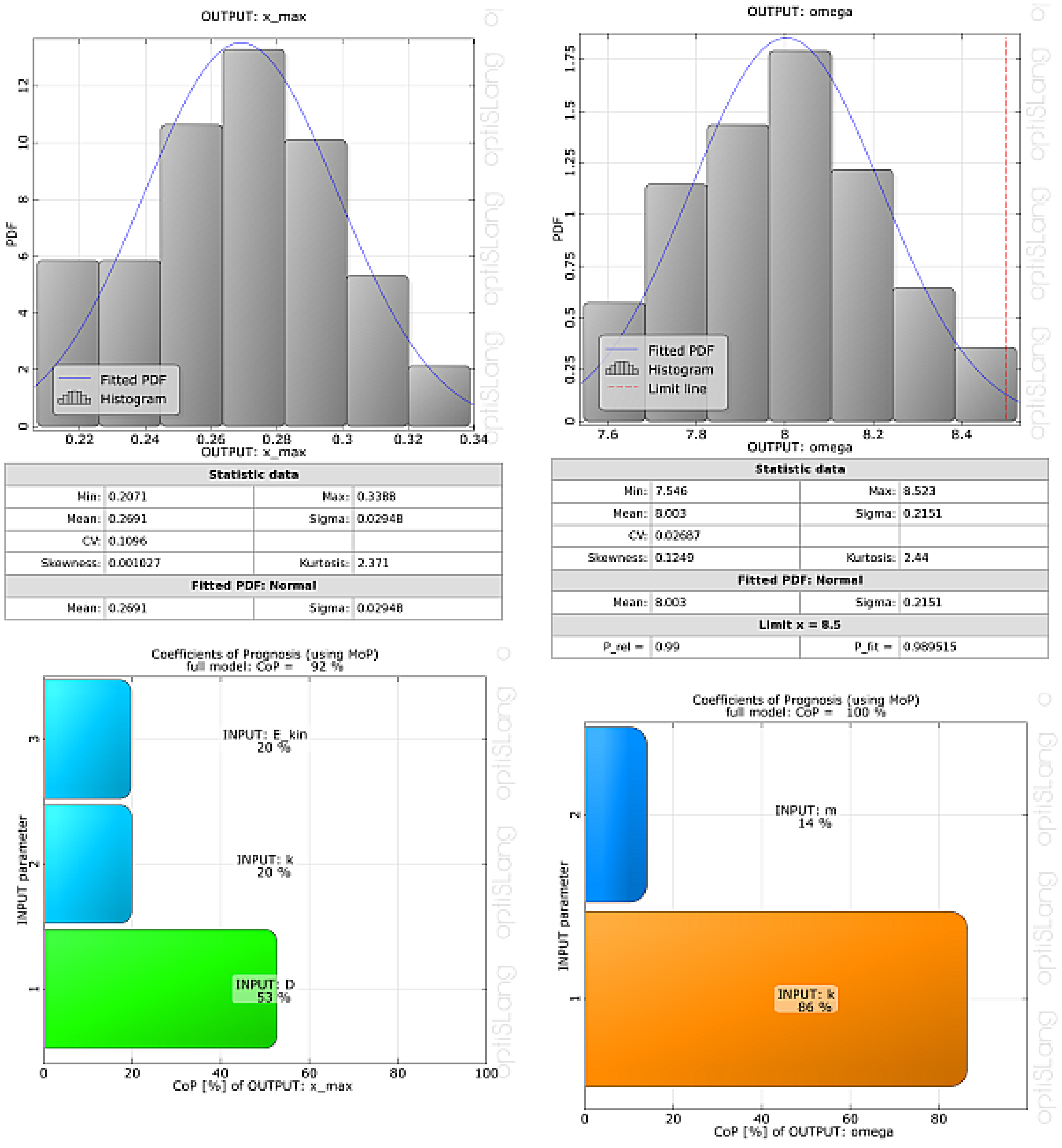
Release 2021 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

in Figure 4.8: Statistical Properties of the Oscillator Responses and Input Sensitivities Obtained with Variance-Based Robustness Analysis (p. 58) can be used to estimate this safety margin, which is only 2.3 sigma. From the viewpoint of the safety requirements, the design found by the deterministic optimization procedures does not fulfill the robustness criteria. In that case, that the input uncertainties could not be reduced, the optimum should be moved back into the safe region until the safety requirements are fulfilled. This strategy is realized in the Robust Design Optimization presented in the next section.

4.3. Robust Design Optimization

In Robust Design Optimization (RDO) a CAE model is optimized under consideration of robustness criteria. In optiSLang an iterative variance-based RDO procedure is available. In this procedure deterministic optimization is utilized by considering safety factors within the constraint conditions. These safety factors should be chosen in that way, that the robustness requirements are fulfilled.

Figure 4.8: Statistical Properties of the Oscillator Responses and Input Sensitivities Obtained with Variance-Based Robustness Analysis

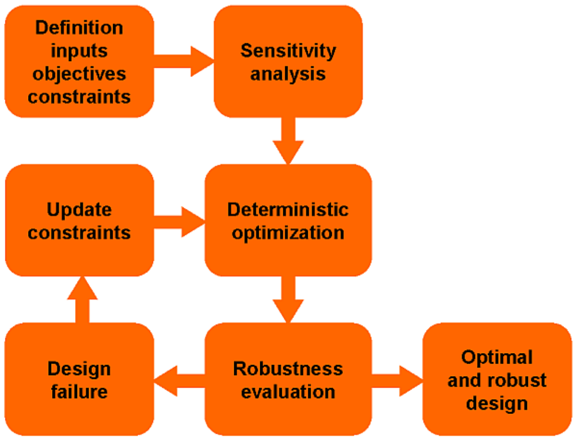


Statistic data			
Min:	0.2071	Max:	0.3388
Mean:	0.2691	Sigma:	0.02948
CV:	0.1096		
Skewness:	0.001027	Kurtosis:	2.371
Fitted PDF: Normal			
Mean:	0.2691	Sigma:	0.02948

Statistic data			
Min:	7.546	Max:	8.523
Mean:	8.003	Sigma:	0.2151
CV:	0.02687		
Skewness:	0.1249	Kurtosis:	2.44
Fitted PDF: Normal			
Mean:	8.003	Sigma:	0.2151
Limit x = 8.5			
P_rel =	0.99	P_fit =	0.989515

Generally the safety factors are not known a priori. In this case the user specifies a suitable initial guess and the initial deterministic optimization is performed. Additionally the robustness criteria are evaluated at the optimal design found by the optimizers. If the robustness requirements are not fulfilled, the optimization constraints are adjusted in a next step and the deterministic optimization procedure and the corresponding robustness analysis are performed again. This procedure is repeated until the robustness requirements are fulfilled. In Figure 4.9: Flowchart of Iterative Variance-Based Robustness Design Optimization with optiSLang (p. 59), the flowchart of the iterative RDO procedure is shown.

Figure 4.9: Flowchart of Iterative Variance-Based Robustness Design Optimization with optiSLang

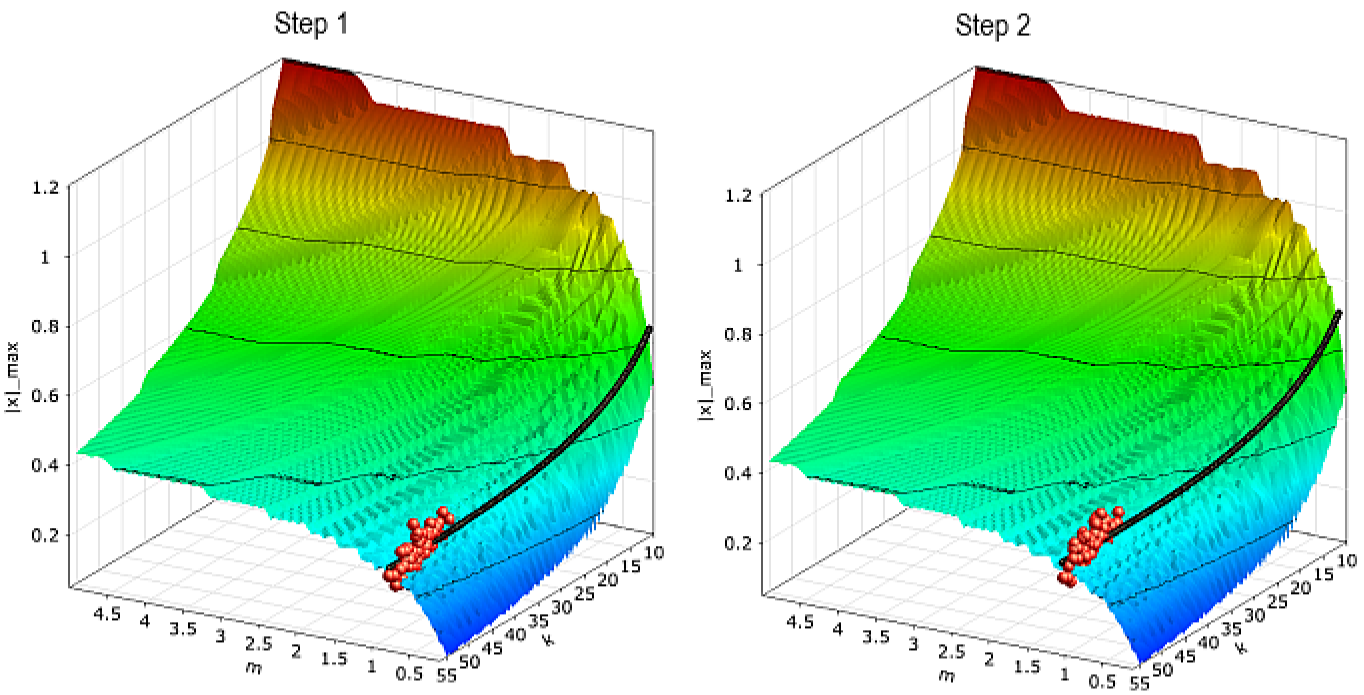

Definition inputs objectives constraints
Sensitivity analysis
Update constraints
Deterministic optimization
Design failure
Robustness evaluation
Optimal and robust design

The iterative RDO procedure is illustrated by means of the oscillator example. Within the initial robustness analysis discussed in Variance-Based Robustness Analysis (p. 53) a safety margin of 2.3σ has been reached. In a next step the optimization constraint is adapted as

$$\omega_{\text{limit}}^{\text{step2}}=8.5-4.5\cdot\sigma_{\omega}^{\text{step1}}=7.6, \tag{4.11}$$

which leads to the optimal design shown in Figure 4.10: Iterative Robust Design Optimization (p. 60). By means of the second robustness analysis using 100 samples the required safety margin of 4.5σ was proven. In case, that two iteration steps are not sufficient, in the third and following steps the constraints values can be linearly interpolated between the values of the previous steps by considering the obtained and the required safety margins. Generally, this procedure ends with an acceptably design within three or four iteration steps. If small failure probabilities are required, the estimate using the safety margin may be very inaccurate. In such a case, a final reliability analysis shall be performed to prove the required probability.

Figure 4.10: Iterative Robust Design Optimization



Optimization (ARSM)						Robustness Analysis (100 LHS)		
	Constraint	*m*	*k*	ω	$\|X\|_{max}$	Mean ω	Sigma ω	Safety Margin
Step1	$\omega \leq 8.0$	0.78	50.0	7.99	0.250	8.00	0.21	2.33σ
Step2	$\omega \leq 7.6$	0.87	49.7	7.55	0.269	7.54	0.20	4.75σ

Iterative Robust Design Optimization of the damped oscillator using 2 steps to obtain 4.5σ safety margin.

Chapter 5: Reliability Analysis

Quality or safety requirements on a product necessitate to take scattering influences into account. These influences can be external (random loads or operation conditions) or inherent in the product itself (like scattering material properties or production tolerances). If the failure rate is small, as it is typical for safety considerations, then the usual framework, which is also established by codes (Eurocode EN 1990 (p. 89)), is the probabilistic safety assessment. The variance-based robustness evaluation, or sigma-level approach, (Equation 4.1 (p. 49)), may not be sufficient, since it is just a one-dimensional consideration and relies on the assumption of normal distribution.

Within the framework of probabilistic safety assessment or reliability analysis, the scattering influences are modelled as random variables, which are defined by distribution type, stochastic moments and mutual correlations, see Definition of Uncertainties (p. 50). The result of the analysis is the complementary of reliability, the probability of failure, which can be represented on a logarithmic scale.

In the following sections, the definition of the probability of failure will be given first. optiSLang offers a variety of methods to compute it, with different ranges of application, which will be introduced then.

5.1. Definition of the Reliability Problem

The first tasks of a reliability analysis are to specify the random input variables with their properties, i.e. distribution types, stochastic moments and correlations (see Definition of Uncertainties (p. 50) and Distribution Types (p. 80)) and to formulate the requirements on the product to be analyzed. The failure criterion does not necessarily indicate collapse of the system, any state of the system that violates quality, serviceability or safety requirements can be formulated as follows: A function of the input random variables *g*(x) scaled such that

$$g(x)\left\{\begin{matrix} <0 \text{ in case of failure} \\ >0 \text{ else} \end{matrix}\right\} \tag{5.1}$$

is called limit state function. *g*(x) = 0 marks the transition from the safe state to the failed state and is called failure surface (Madsen, Krenk, and Lind 1986 (p. 90)). Often the limit state function cannot be formulated directly depending on input random variables, but is implicitly given by the result of a simulation of the examined system.

This is called an implicit limit state. A typical example is *g(r, s)* = *r -s* with *r*, the resistance (ultimate stress) and s, computed stress of a structure.

With this definition, the space of all involved variables $\mathbb{R}^n$ can be separated into the safe domain and the failure domain

$$\mathbb{D}_F = \{ \mathrm{x} \in \mathbb{R}^n | g(\mathrm{x}) < 0 \} \tag{5.2}$$

The probability of failure P_f is the probability of the event that x falls into the failure domain.

$$\begin{aligned} P_f &= P[X \colon X \in \mathbb{D}_F] \\ &= P[g(X) < 0] \end{aligned} \tag{5.3}$$

and can be computed as the integral of the joint density of all random variables over the failure domain $\mathbb{D}_F$

$$P_f = \int \underset{\mathbb{D}_F}{\cdots} \int f\mathrm{x}(\mathrm{x})\ dx \tag{5.4}$$

Figure 5.1: Reliability Problem in Two Dimensions (p. 62) helps to visualize these definitions for the case of two random variables.

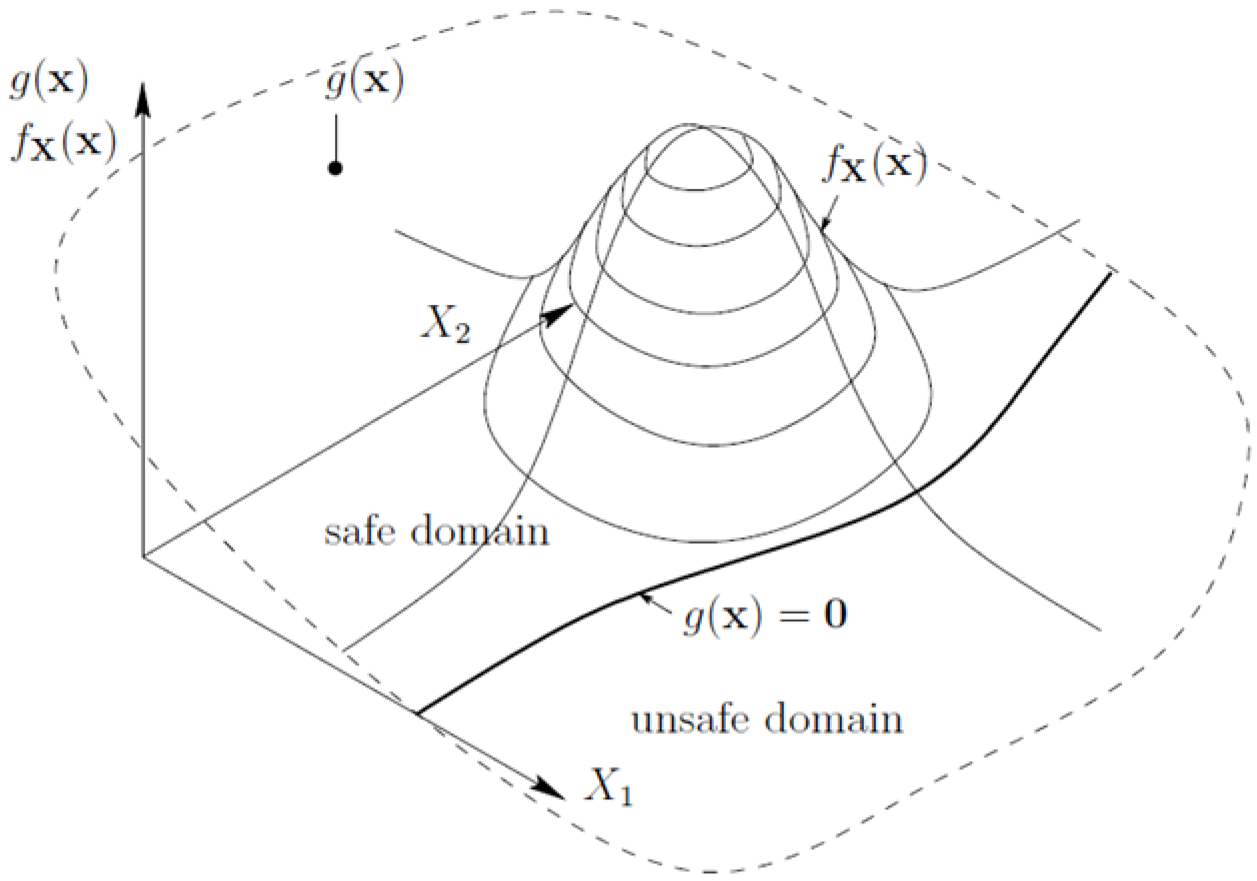
The limit state function may comprise several failure criteria. Two basic cases are demonstrated in the following. A system consisting of several components which fails entirely, if just one of its components fails, is called a series system or weakest link system. (Think of a chain under tension.) The probability of failure for the case of independent single failure modes is

$$P_f = \bigcup_{j=1}^{m} P_{fi} \tag{5.5}$$

and the limit state function is computed as

$$g(\mathrm{x}) = \min_{j=1..m} (g_j(\mathrm{x})) \tag{5.6}$$

Figure 5.1: Reliability Problem in Two Dimensions



Sketch of the reliability problem in two dimensions. f_X(x): joint prob- ability density, g(x): limit state function, g(x) = 0: limit state surface.

A parallel system, in contrast, has redundancy. The system fails when all components have failed. (E.g., a wire rope.) Then

$$P_f = \bigcap_{j=1}^{m} P_{fi} \tag{5.7}$$

$$g(\mathrm{x}) = \max_{j=1..m} (g_j(\mathrm{x})) \tag{5.8}$$

When setting up a reliability analysis in optiSLang, always the safe case, $g(\mathrm{x}) > 0$ is formulated. If several criteria are given by several lines in the limit states table, they are automatically interpreted as series system.

5.2. Standardization and Generation of Random Numbers

In particular for problems involving more than one random variable, and/or different failure criteria which are considered simultaneously, a proper scaling of the variables is important for numerical stability of the computations. By the linear transformation

$$Y = \frac{X - \mu \mathrm{x}}{\sigma \mathrm{x}} \tag{5.9}$$

with μ_X and σ_X, the mean and standard deviation of *X*, respectively, *Y* has mean value zero and standard deviation one. For a random vector **X** of possibly correlated random variables, the covariance matrix $\mathbf{C_{XX}}$ has to be decomposed to the form $\mathbf{C_{XX}} = \mathbf{LL}^T$ e.g. by the Cholesky algorithm. Then

$$\mathbf{Y} = \mathbf{L}^{-1}(\mathbf{X} - \mu_{\mathrm{x}}) \tag{5.10}$$

is a random vector with zero means, unit standard deviations and zero correlations.

Many reliability algorithms are based on standardized normal random variables **U**. These are produced by the algorithm and have to be transformed back to the real-world variables **X** in order to evaluate the simulated system with the random input. For this purpose, the marginal distributions and correlations are defined and the Nataf model, as introduced in Multivariate Distributions (p. 52), is established in advance. Then by applying the inverse standardization on a realization **u** and then the inverse Nataf transformation, the sample **x** in original space is obtained.

5.3. First Order Reliability Method (FORM)

The First Order Reliability Method or FORM (see *The Hasofer and Lind Reliability Index* in Madsen, Krenk, and Lind 1986 (p. 90)) is a well established method for reliability calculation (Eurocode EN 1990 (p. 89)). It provides an analytical solution based on assumptions and simplifications. Therefore, it fast to compute, but in special cases it may be not accurate. The method provides no measure of accuracy (such as, e.g. a confidence level).

The FORM result is defined in the space of standard normal variables **U**, thus suitable transformations are required, see the previous section. If the mean vector μ_X is transformed such, it coincides with the origin of the **U**-space.

Figure 5.2: Linearisation at the Design Point

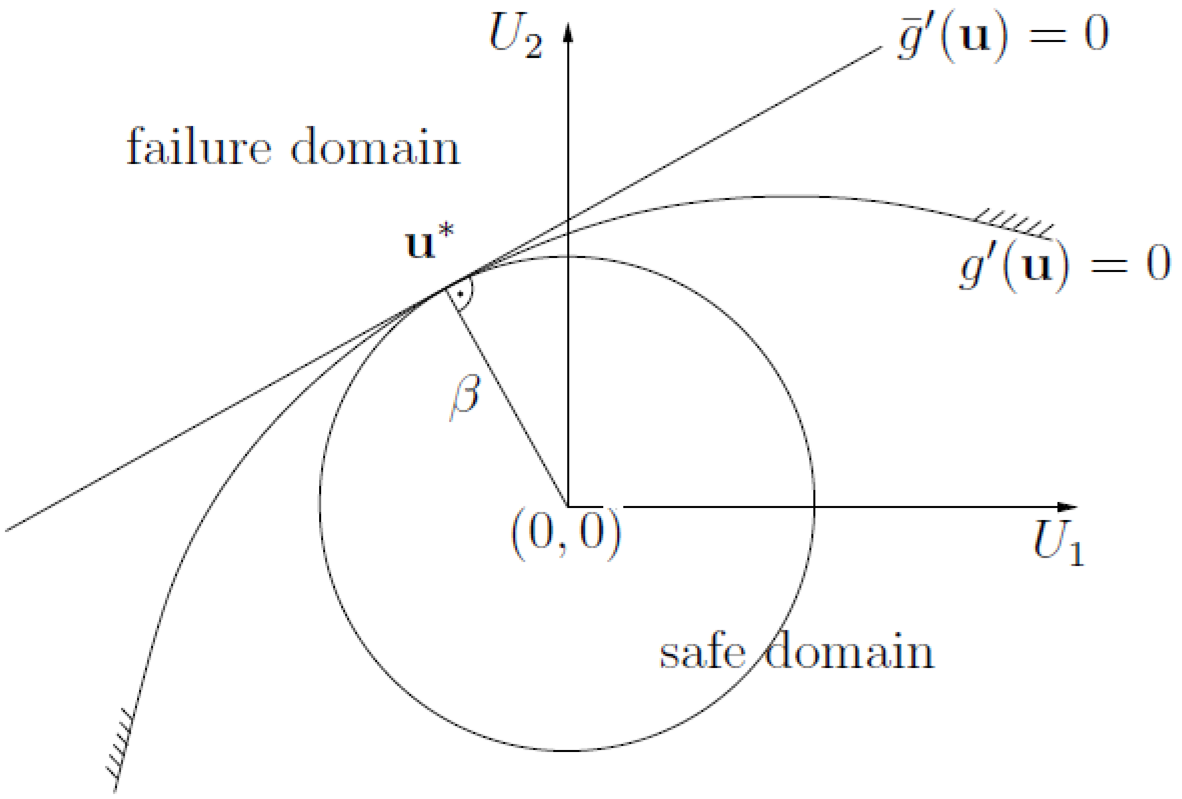


Table 5.1: Values β For Different Probabilities of Failure

P_f	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}
β	1.28	2.33	3.09	3.72	4.26	4.75	5.20

The geometrical distance from the origin of the **U**-space to any point on the transformed failure surface *g*'(**u**) = 0 corresponds to the respective distance between mean vector and failure surface in original space, as expressed by multiples of standard deviations. The point on the failure surface with largest probability density corresponds to the smallest distance, which is denoted as the reliability index

$$\beta = \min_{u|g'(u)=o} (u^T u)^{1/2} \tag{5.11}$$

Figure 5.2: Linearisation at the Design Point (p. 64) illustrates this geometrical interpretation. Hence the algorithm has to find the point with minimal distance to the origin of U-space. This is the so-called design point **u** *. It can be found by any optimization algorithm (Bucher 2009 (p. 89)).

If the limit state function is linearized at the design point, the corresponding probability of failure can be computed as

$$\bar{P}_f = \Phi(-\beta) \tag{5.12}$$

where $\Phi(\,)$ is the standard normal cumulative distribution function. This solution is exact only for the special case that the failure surface is linear in **U**-space.

Release 2021 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The success of the method depends largely on the success of the optimization algorithm, so the same requirements as for optimization tasks hold. If a gradient-based optimizer is applied, the limit state function (in **U**-space) must be continuous and sufficiently smooth. The design point must be unique. If the failure surface is not linear at the design point, then the failure probability computed by linearization may become inaccurate. However, this inaccuracy is overlaid by other sources of uncertainty and diminishes to some extent with increasing distance.

Generalized Reliability Index

If the probability of failure is computed by any other method than FORM, the reliability index can be obtained from inversion of Equation 5.12 (p. 64). This so-called generalized reliability index is a scaled representation of the probability of failure. See (Madsen, Krenk, and Lind 1986 (p. 90)) for further considerations on this topic. Table 5.1: Values β For Different Probabilities of Failure (p. 64) lists typical values of the relation β vs. P_f .

5.4. Monte Carlo Simulation

An indicator function is introduced into Equation 5.4 (p. 62), which adopts the value one if the system, examined for a random parameter set, is in a state considered as failure,and zero else. Formally:

$$I(g(\mathrm{x}))=\begin{cases}0 \text{ if } g(\mathrm{x})>0\\ 1 \text{ if } g(\mathrm{x})\leq 0\end{cases} \tag{5.13}$$

After setting this indicator into the integral over the failure domain, the integration domain can be changed without influencing the value of the integral. The resulting notation suggests that the integral can be interpreted as expected value of the indicator $I(g(\mathbf{x}))$.

$$\begin{aligned} P_f &= \int \underset{\mathbb{D}_F}{\cdots} \int f_{\mathrm{X}}(\mathrm{x})\ d\mathrm{x} \\ &= \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} I(g(\mathrm{x}))\ f_{\mathrm{X}}(\mathrm{x})\ d\mathrm{x} \\ &= \mathrm{E}[\mathrm{I}(g(\mathrm{x}))] \end{aligned} \tag{5.14}$$

The concept of Monte Carlo simulation is simply to generate a sample of random vectors x_i , submit each sample to the simulation of the examined system as input parameter set and compute the limit state function. Then the average of the sample of indicator values is computed as an estimator of the expected value and therefore of the probability of failure.

$$\bar{P}_f=\frac{1}{m}\sum_{i=1}^{m} I(g(\mathrm{x}_i)) \tag{5.15}$$

In the above, x_i is the i th realisation in a sample of size m. It can be proven that $\bar{P}_f$ is an unbiased estimator for the probability of failure.

An estimator which depends on random inputs is a random variable itself. The variance of this estimator is defined as (Rubinstein 1981 (p. 90))

$$\sigma^2_{\bar{P}_f}=\frac{P_f-P_f^2}{m} \tag{5.16}$$

It can be computed approximately from the sample and will approach zero for $m \rightarrow \infty$.

The smaller the variance, the more trustworthy is the estimate. The estimator standard deviation normalized to the sample size,

$$S_{\bar{P}_f} = \sqrt{\frac{\sigma^2_{\bar{P}_f}}{m}} \tag{5.17}$$

is also referred to as standard error.

The plain Monte Carlo method is the most general and robust way to compute the probability of failure, yet an inefficient one. The required sample size (and thus the number of solver calls) does not depend on the dimension, but on the expected failure probability. If, for example, a failure probability of $P_f = 10^{-3}$ shall be computed with a coefficient of variation of 10%, a required sample of size $m = 10^5$ is derived from Equation 5.16 (p. 65).

5.5. Importance Sampling

Since the variance of the estimator for P_f corresponds to its confidence, variance reduction techniques aim at influencing the sampling such that the estimator variance becomes smaller, the confidence intervall narrower, hence the estimate of P_f becomes more accurate. A widely used technique is the Importance Sampling. The principle is to guide the sampling with any a priori information, such that the ratio of failure events to the total sample size increases. This of course is an intentional influence of the statistics, which has to be corrected to warrant unbiasedness of the estimator.

Samples are not generated following the prescribed density $f_{\mathbf{X}}$, but with a sampling density denoted as h **Y**. Setting h **Y**/h **Y** into the integral (Equation 5.14 (p. 65)) does not change its value, but the integral can be interpreted as expected value within a population **Y**.

$$\begin{aligned} P_f &= \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} I(g(y)) \frac{f_X(y)}{h_Y(y)} h_Y(y)\; dy \\ &= \mathrm{E}\left[I(g(y)) \frac{f_X(y)}{h_Y(y)} \right] \end{aligned} \tag{5.18}$$

The estimator now includes the correction term, also called importance sampling weight:

$$\bar{P}_f = \frac{1}{m} \sum_{i=1}^{m} I(g(y_i)) \frac{f_X(y_i)}{h_Y(y_i)} \tag{5.19}$$

This estimator can also be proved to be unbiased and consistent (Rubinstein 1981 (p. 90)).

5.5.1. Adaptive Sampling (ADSAP)

Many variants have been developed which shall on one hand reduce the estimator variance most efficiently, on the other hand be robust and versatile for a large range of applications. One such Importance Sampling technique is the Adaptive Sampling by Bucher, 1988 (see Bucher 2009 (p. 89)).

The procedure involves several simulation runs. For the first step, the sampling density may have larger scatter than originally defined for the input parameters. The samples which fell into the failure domain $\mathbb{D}_F : \mathrm{X} | g(\mathrm{x}) \leq 0$ in the first run are statistically evaluated: The result serves to define a multidimensional normal type simulation density hY(**Y**) for the subsequent importance sampling run:

$$E[\mathbf{Y}] = E[\mathbf{X} | g(\mathbf{x}) \leq 0] \tag{5.20}$$

Release 2021 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

$$E[\mathbf{YY}^T] = E[\mathbf{XX}^T | g(\mathbf{x}) \leq 0] \tag{5.21}$$

A third run (that is, a second adaptation) should be performed to prove stability of the result.

A theoretically 'ideal' sampling density which would reduce the estimator variance to zero (Rubinstein 1981 (p. 90)) is approximated in the second moment sense by this procedure. The Adpative Sampling technique has a large range of applications, including not-differentiable and noisy limit state functions. Since the effort required to compute a covariance matrix increases quadratically with dimension n, it may become ineffective in high dimensions. If the user has an idea about the expected reliability index, it is recommended to use this as a scaling factor of the sampling standard deviation in the first run.

Example: Parabola

The properties of the Adaptive Sampling method shall be illuminated by a simple example. The random variables are

$$X_1 \sim N(0;1)$$
$$X_2 \sim N(3;1)$$

and the limit state is a parabola:

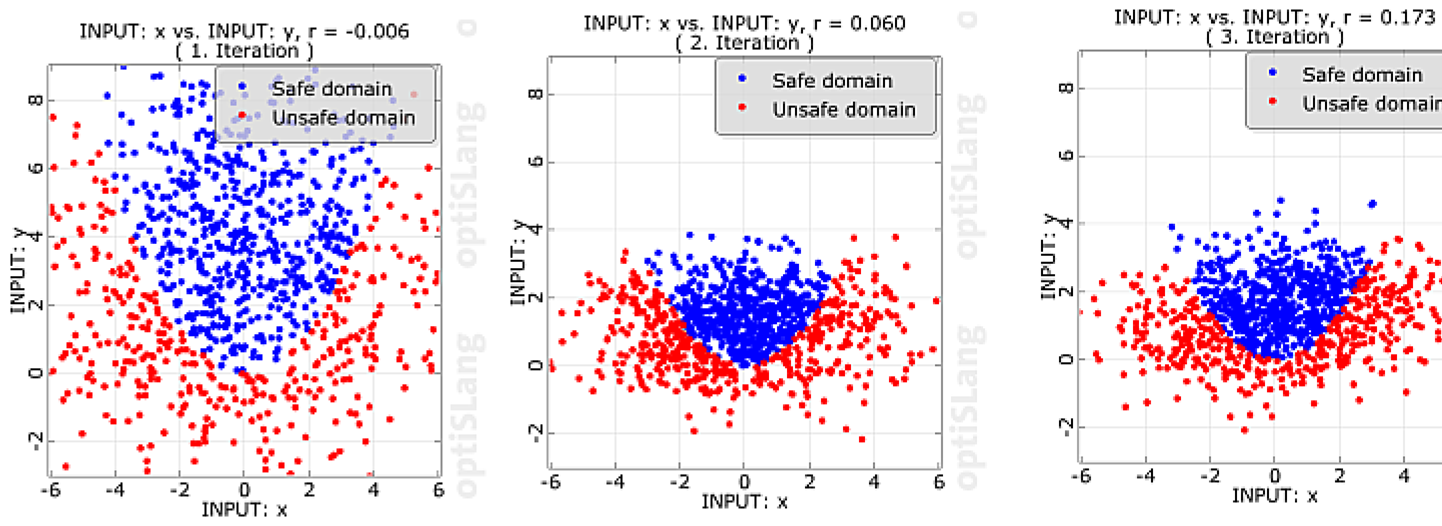
$$g(\mathbf{x}) = x_2 - x_1^2/3$$

The limit state function is smoothly non-linear, the design point is not unique. For the Adaptive Sampling procedure in optiSLang, three runs with 1000 samples each were performed. The standard deviations were scaled by a factor of three in the first run. Figure 5.3: Parabola Example: Anthill Plots of Subsequent Adaptive Sampling Runs (p. 68) shows how the sampling densities are adapted in the subsequent runs.

5.5.2. Importance Sampling Using the Design Point (ISPUD)

The ISPUD strategy (Bourgund and Bucher 1986 (p. 89)) is to center the sampling density at the failure surface in the region of highest probability density. For this purpose, a searching procedure analogous to FORM (First Order Reliability Method (FORM) (p. 63)) is applied.

Figure 5.3: Parabola Example: Anthill Plots of Subsequent Adaptive Sampling Runs



The sampling density $h_{\mathbf{Y}}$ has a mean vector defined by the design point transformed back to original space. Distribution types, variances and correlations are taken over from the definitions of the original random parameters.

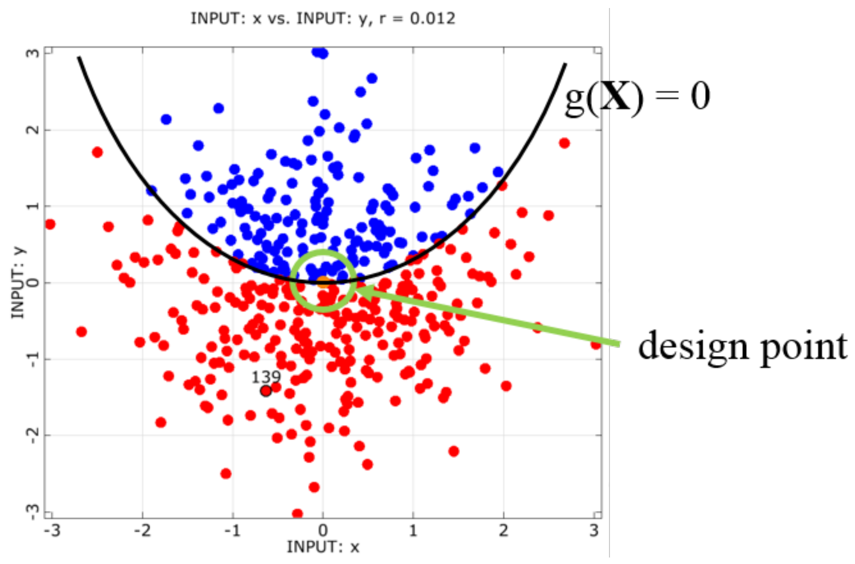
$$E[\mathbf{X}] = \mathbf{u}^* \tag{5.22}$$

$$\mathbf{C}_{YY} = \mathbf{C}_{XX} \tag{5.23}$$

Figure 5.4: Parabola Example: Anthill Plot of Importance Sampling Using the Design Point (p. 68) shows the ISPUD sampling for the same example of a parabola, as it is introduced in Adaptive Sampling (ADSAP) (p. 66).

As for FORM, the success of the optimization step is crucial also for this method. At present, the gradient-based NLPQL algorithm is used, so the limit state function has to be sufficiently smooth, continuous, with a unique design point. If the design point can be determined correctly, ISPUD can overcome inaccuracies due to the linearization, which is typical for FORM.

Figure 5.4: Parabola Example: Anthill Plot of Importance Sampling Using the Design Point



Release 2021 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

5.6. Directional Sampling (DS)

The variance of the estimator for the probability of failure can be reduced by in troducing analytical partial solutions. This is done in the Directional Sampling procedure (Deák 1980 (p. 89)), (Bjerager 1988 (p. 89)). The original random variables are trans- formed into the space of standard Gaussian variables. A random vector in standard Gaussian space is represented in polar coordinates

$$\mathrm{U} = \mathrm{R} \cdot \mathrm{A} \tag{5.24}$$

with **A** being a unit vector, and *R* the Euclidian norm of **U**. With this, Equation 5.4 (p. 62) can be re-written as

$$P_f = \int \underset{\mathbb{D}_F}{\cdots} \int f_{R,\mathbf{A}}(r, \mathbf{a})\, dr d\mathbf{a} \tag{5.25}$$

Due to the transformation of the standard Gaussian variables from Cartesian into polar coordinates, the unit vectors **A** are uniformly distributed over the surface of a hypersphere with radius 1, centered at the origin. The radii *R* are independent of **A** and follow a X^2-distribution with *k* degrees of freedom, *k* being the dimension.

The procedure is to generate unit vectors **a** $_i$, for each of which the distance of the failure surface to the origin $\mathbf{r}_i^*$ is found by a suitable iteration. Then the conditional probability of failure given a direction ai is computed as

$$P_f | \mathbf{a}_i = 1 - \chi_k^2 (r_i^{*2} (\mathbf{a}_i)) \tag{5.26}$$

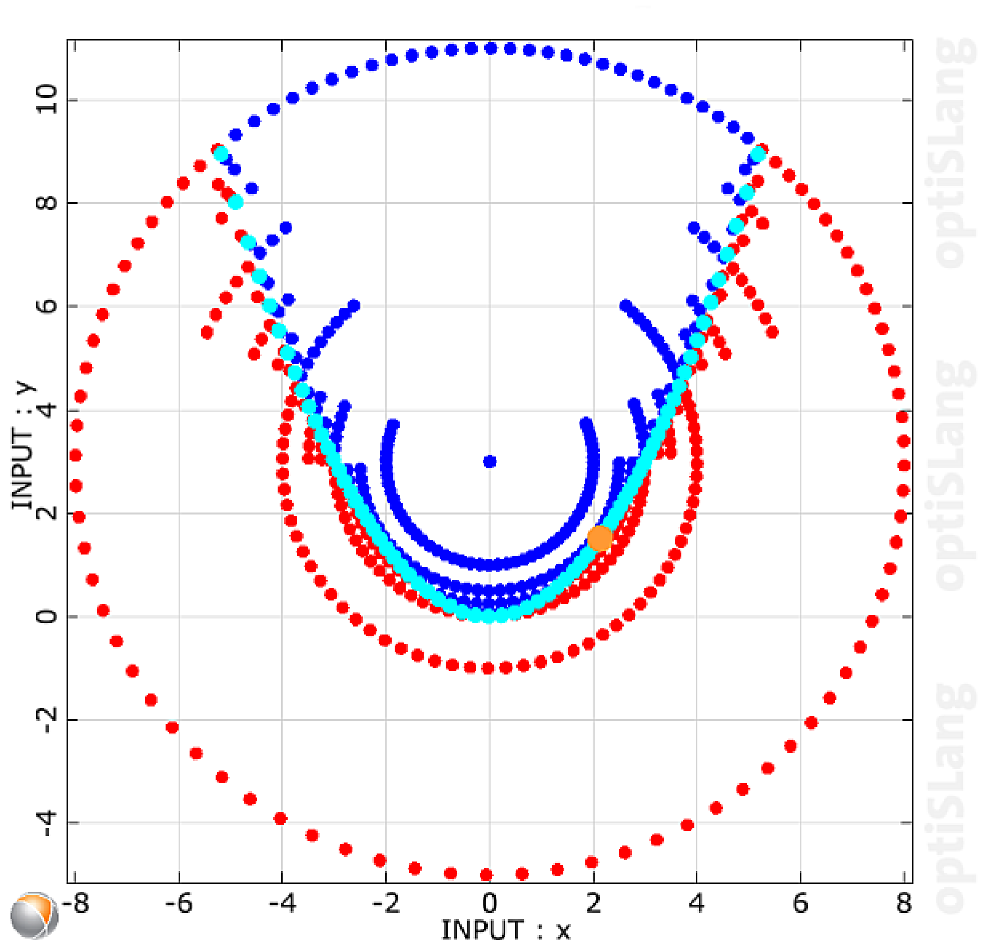
and the total failure probability is the mean of the conditional failure probabilities with respect to the random vectors **A**.

$$P_f = E_{\mathbf{A}} [P_f | \mathbf{A}] \tag{5.27}$$

$$\approx \frac{1}{n} \sum_{i=1}^{n} P_f | a_i \tag{5.28}$$

A basic prerequisite for Directional Sampling is, that the mean vector of the original random variables $\overline{X}$ has to lie within the safe domain, and the limit surface in each direction shall be unique. Failure should be formulated in a way that the iteration is able to operate in the failure domain (no premature exit from the solver). However, because of the monotonic decrease with increasing distance to the mean vector of the joint density function, a non-unique failure surface may be acceptable, as long as the closest root of $g'(r, \mathrm{a}_i) = 0$ is found.

Figure 5.5: Parabola Example: Anthill Plot of Directional Sampling Procedure



The Directional Sampling method is sensitive towards dimension. It does not put any requirements on smoothness nor continuity of the limit state function. Moreover, the implementation in optiSLang is able to handle a moderate number of unsuccessful solver calls.

Figure 5.5: Parabola Example: Anthill Plot of Directional Sampling Procedure (p. 70) shows the Directional Sampling results for the parabola example introduced in Adaptive Sampling (ADSAP) (p. 66). A bisection algorithm is used to iterate the failure surface. The iteration does not exceed a distance to the mean vector of eight standard deviations.

5.7. Adaptive Response Surface Method (ARSM-DS) for Reliability Analysis

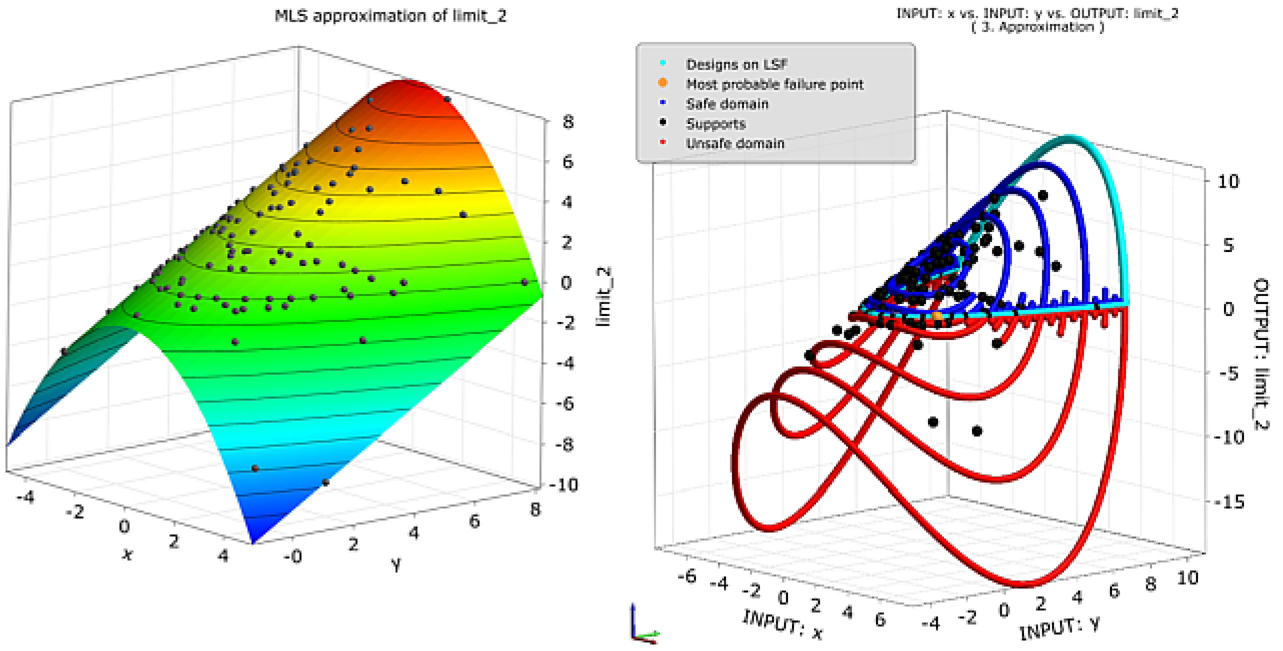
Like in optimization, it seems attractive to substitute the solver responses by a surrogate model in order to save computation time for the solver calls in the reliability analysis, or to afford more function evaluations of the surrogate. It should be mentioned that accuracy requirements are higher in reliability analysis as compared to optimization. Early approaches (Faravelli 1989 (p. 89)), (Bucher and Bourgund 1990 (p. 89)) are based on classical Response Surface Methodology (RSM) using systematic designs of experiments (DoE) and polynomial regression models.

In the current optiSLang implementation, supports are generated by Latin Hypercube Sampling in the first step. The approximation model is of Moving Least Squares type, see Metamodel of Optimal Pro-

Release 2021 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

gnosis (p. 18). The probability of failure is computed by Directional Sampling, applied on the surrogate model.

Figure 5.6: Parabola Example: Response Surface and Directional Sampling Generated by Adaptive Response Surface Procedure



Directional Sampling yields points on the failure surface of the surrogate model $(x_0)=0$. It is assumed that these points are already close to the "true" failure surface. Hence in the subsequent steps, these points are updated by passing the input values to the solver and computing the respective responses. The flexibility of the Moving Least Squares model does not require regular DoE schemes as supports and allows for a local improvement in the region of interest, i.e. close to the failure surface.

The accuracy of this method depends largely on the quality of the surrogate model. It is measured as the R^2 *predicted* value computed by a "leave one out" algorithm. The Adaptive Response Surface method proves to be very efficient and accurate for a wide range of applications, but typically, the efficiency decreases with increasing dimension.

??? (p. 71) shows the moving least squares model and an anthill plot for the parabola example (Adaptive Sampling (ADSAP) (p. 66)). The anthill plot comprises supports of the response surface and samples of the directional sampling procedure. The additional supports can be observed near the "Designs on Limit State Surface".

5.8. Recommendations for Robustness Wizard

Table 5.2: Range of Application of Different Qualified Reliability Methods (p. 72) provides an overview of reliability algorithms implemented in optiSLang along with some recommendations for application.

Table 5.2: Range of Application of Different Qualified Reliability Methods

Approach	Non-Linearity	Failure Domains	Number of Paramenters	Number of Solver Runs
Monte Carlo SImulation	arbitrary	arbitrary	many	>10^4 (3 sigma) >10^7 (5 sigma)
Directional Sampling	arbitrary	arbitrary	<= 10	1000-5000
Adaptive Importance Sampling	arbitrary	one dominant	<= 10	500-1000
FORM, SORM, ISPUD	monotonic	one dominant	<= 20	200-500
Adaptive Response Surface Method	continuous	few dominant	<= 20	200-500

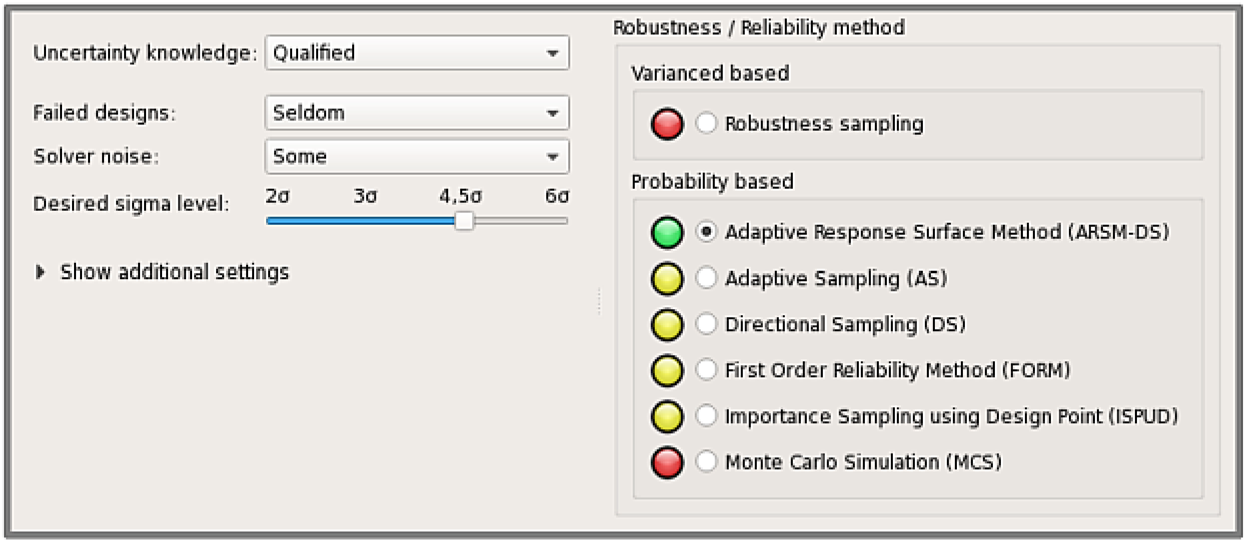
The robustness wizard in optiSLang helps with the decision of which method to choose. If the robustness analysis is set up with help of this wizard, there is a dialog in which some information has to be provided by the user as follows:

- **Uncertainty knowledge**: Unaware | Estimated | Qualified
- **Failed designs**: None | Seldom | Frequently
- **Solver noise**: None | Some | Strong
- **Sigma level**: Range $2\sigma - 6\sigma$ set by slider

The number of stochastic parameters and defined limit states is already known to the program. The response is given as a traffic light in the list of algorithms to choose from, see Figure 5.7: Dialog of the Robustness Wizard (p. 73). The green color marks the recommended method, red is discouraged. A method marked yellow can be chosen by the experienced user, this should be done with care. If no limit state is defined and/or the uncertainty knowledge is unaware or estimated, then the recommendation is "Robustness sampling", i.e. a variance based robustness analysis.

Release 2021 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 5.7: Dialog of the Robustness Wizard



Uncertainty knowledge: Qualified

Failed designs: Seldom

Solver noise: Some

Desired sigma level: 2σ 3σ 4,5σ 6σ

Show additional settings

Robustness / Reliability method

Varianced based

🟥 ○ Robustness sampling

Probability based

🟩 ● Adaptive Response Surface Method (ARSM-DS)

🟨 ○ Adaptive Sampling (AS)

🟨 ○ Directional Sampling (DS)

🟨 ○ First Order Reliability Method (FORM)

🟨 ○ Importance Sampling using Design Point (ISPUD)

🟥 ○ Monte Carlo Simulation (MCS)

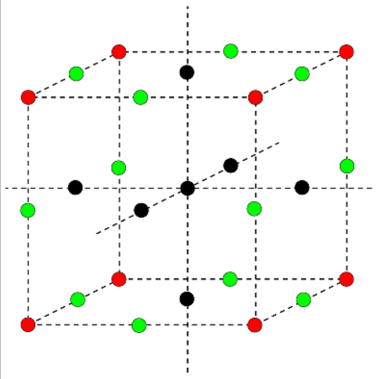
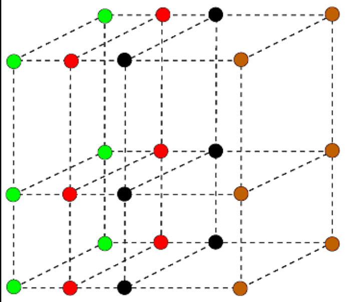
Dialog of the robustness wizard: recommendation of an algorithm dependent on characterization of the robustness task.

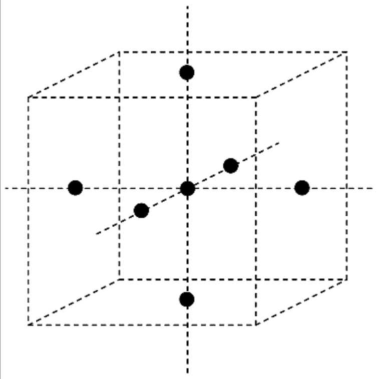
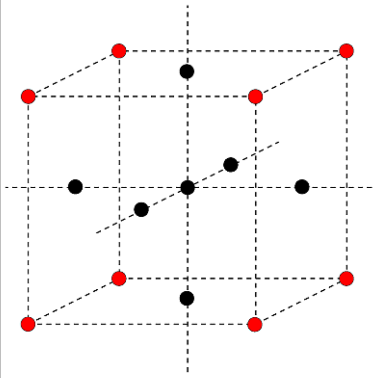
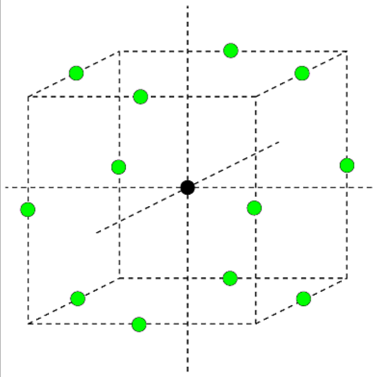
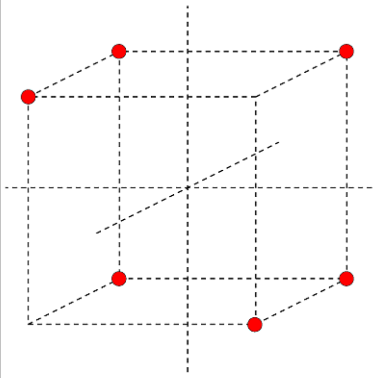
Release 2021 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information
of ANSYS, Inc. and its subsidiaries and affiliates.

Appendix A. DOE Schemes

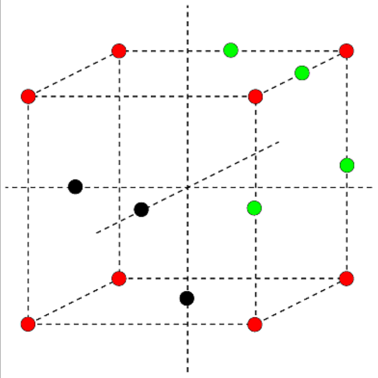
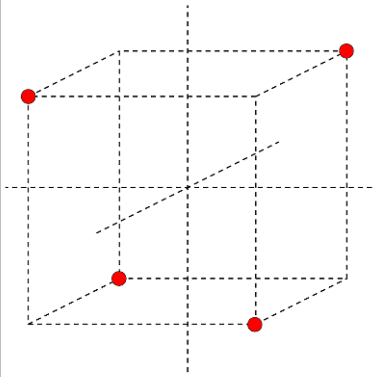
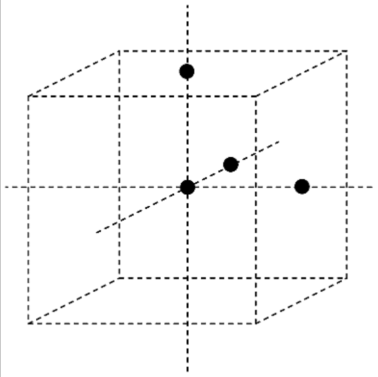
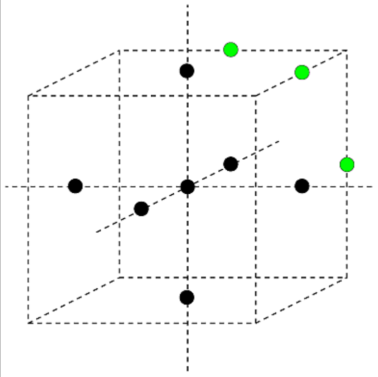
Deterministic DOE Types

Table 1: Deterministic Design of Experiment

Name	Image	Description
Full factorial		• Number of input parameter: $k \geq 1$ (continuous and discrete) • Number of levels: $p \geq 2$ • Number of samples: $n = p^k$ • Linear terms • Mixed terms • Quadratic terms if $p \geq 3$
Full Combinatorial		This DOE generates all possible combinations of discrete input values • Number of input parameter: $k \geq 1$ • Number of discrete values p_i for each dimension X_i • Number of samples: $n=\prod_{i=1}^{k} p_i$ • Linear terms • Mixed terms • Quadratic terms if all $p_i \geq 3$

Name	Image	Description
Star points		• Number of input parameter: $k \geq 1$ (continuous and discrete) • Number of levels: $p \geq 1$ • Number of samples: $n = 1 + 2pk$ • Linear terms • Quadratic terms
Central composite		• Number of input parameter: $k \geq 2$ (continuous and discrete with $\geq$ 3 values) • Number of samples: $n = 1 + 2k + 2^k$ • Linear terms • Mixed terms • Quadratic terms
Box-Behnken		• Number of input parameter: $k \geq 3$ (continuous and discrete with $\geq$ 3 values) • Number of samples: $n = 2k(k-1) + 1$ • Linear terms • Mixed terms • Quadratic terms
D-optimal linear		• Number of input parameter: $k \geq 1$ (continuous and discrete) • Number of samples: $n=\frac{3}{2}(1+k)$ • Linear terms

Release 2021 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Name	Image	Description
D-optimal quadratic		• Number of input parameter: $k \geq 3$ (continuous and discrete with $\geq$ 3 values) • Number of samples: $n=\frac{3}{2}(1+k+\frac{1}{2}k(k+1))$ • Linear terms • Mixed terms • Quadratic terms
D-optimal customized		• Number of input parameter: $k \geq 1$ (continuous and discrete) • Number of samples: $n=1+k\leq n\leq 2^k$ • Linear terms
Koshal linear		• Number of input parameter: $k \geq 1$ (continuous and discrete) • Number of samples: $n = 1 + k$ • Linear terms
Koshal quadratic		• Number of input parameter: $k \geq 1$ (continuous and discrete with $\geq$ 3 values) • Number of samples: $n=1+k+\frac{1}{2}k(k+1)$ • Linear terms • Mixed terms • Quadratic terms

Release 2021 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Appendix B. Random Variables

See the following sections:

B.1. Statistical Moments and Process Capability

Table 1: Statistical Moments of a Random Variable

Moment	Expected Value	Estimator
Mean value	$\overline{X}=E[X]=\int_{-\infty}^{\infty} x f_X(x)dx$	$\hat{m}_X=\frac{1}{n}\sum_{i=1}^{n} x_i$
Standard deviation	$\sigma_X=\sqrt{E[(X-\overline{X})^2]}$	$\hat{s}_X=\sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i-\hat{m}_X)^2}$
Skewness	$\gamma 1=\frac{E[(X-\overline{X})^3]}{\sigma_X^3}$	$\hat{g}1=\frac{n}{(n-1)(n-2)}\sum_{i=1}^{n}\frac{(x_i-\hat{m}_X)^3}{\hat{s}_X^3}$
Excess kurtosis	$\gamma 2=\frac{E[(X-\overline{X})^4]}{\sigma_X^4}-3$	$\hat{g}2=\frac{n(n+1)}{(n-1)(n-2)(n-3)}\sum_{i=1}^{n}\frac{(x_i-\hat{m}_X)^4}{\hat{s}_X^4}-3\frac{(n-1)^2}{(n-2)(n-3)}$

Statistical moments of a random variable: definition by expected values and statistical estimators for a given sample set.

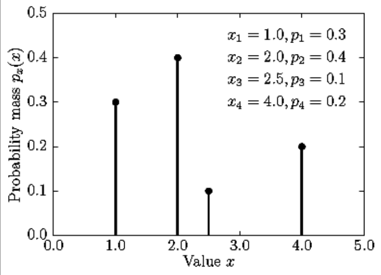
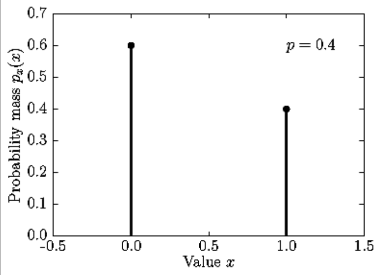
Table 2: Process Capability Indices

$\hat{C}_p=\frac{USL-LSL}{6\hat{s}_X}$	$\hat{C}_{p,lower}=\frac{\hat{m}_X-LSL}{3\hat{s}_X}$	$\hat{C}_{p,upper}=\frac{USL-\hat{m}_X}{3\hat{s}_X}$
$\hat{C}_{pk}=\min[\hat{C}_{p,lower},\hat{C}_{p,upper}]$	$\hat{C}_{pm}=\frac{\hat{C}_p}{\sqrt{1+\left(\frac{\hat{m}_X-T}{\hat{s}_X}\right)^2}}$	$\hat{C}_{pkm}=\frac{\hat{C}_{pk}}{\sqrt{1+\left(\frac{\hat{m}_X-T}{\hat{s}_X}\right)^2}}$

Process capability indices considering the lower specification limit *LSL*, upper specification limit *USL*, the target value *T* as well as the sample mean $\hat{\mu}_X$ and sample standard deviation $\hat{s}_X$.

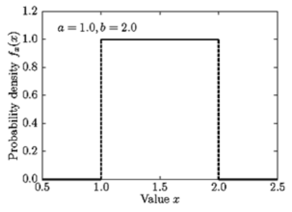
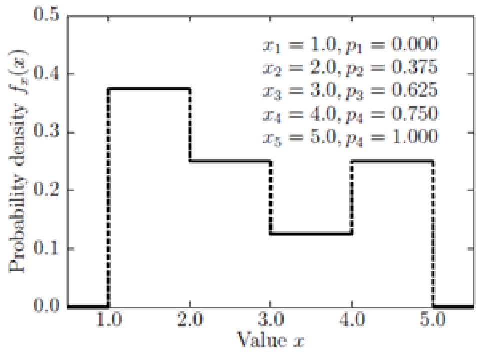
B.2. Distribution Types

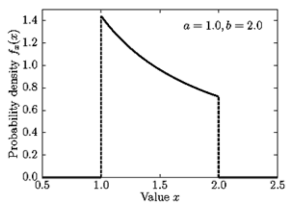
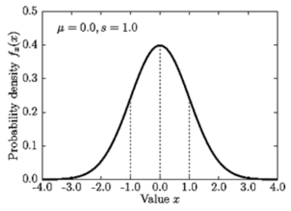
Table 3: Discrete Distribution Types

Name	Image	Description
Discrete distribution	 $x \in \{x_1, \ldots, x_n\},\ 0<p_i<1$	Distribution parameter • *n* pairs of: • Value x_i • Probability $p_i>0,\ \sum_{i=1}^{n} p_i=1$ Mean value and standard deviation • $\overline{X}=\sum_{i=1}^{n} p_i x_i$ • $\sigma_X=\sqrt{\sum_{i=1}^{n} p_i (x_i-\overline{X})^2}$ Probability mass function • $p_x(x)=\begin{cases}1-p \text{ if } x=x_i,\ i=1,\ldots,n \\ 0 \quad \text{otherwise}\end{cases}$
Bernoulli distribution	 $x \in \{0,1\},\ 0<p<1$	Distribution parameter • Success probability 0 < *p* < 1 Mean value and standard deviation • $\overline{X}=p$ • $\sigma_X=p(1-p)$ Probability mass function • $p_x(x)=\begin{cases}1-p \text{ if } x=0 \\ p \quad \text{if } x=1 \\ 0 \quad \text{otherwise}\end{cases}$

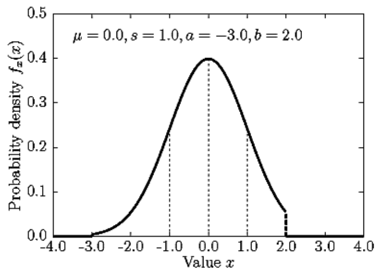
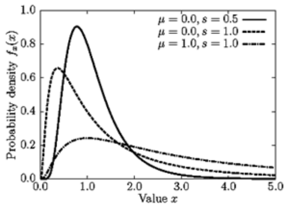
Discrete distribution types: distribution parameters, probability mass function $p_X(x)$, mean value $\overline{X}$, and standard deveation σ_X.

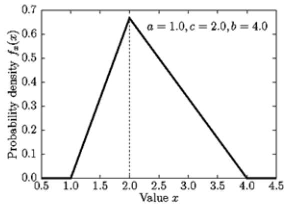
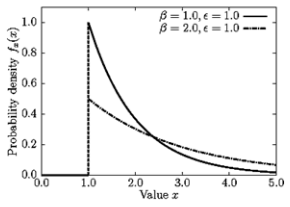
Table 4: Continuous Distribution Types

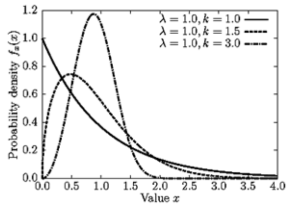
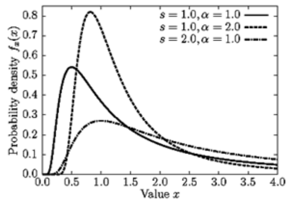
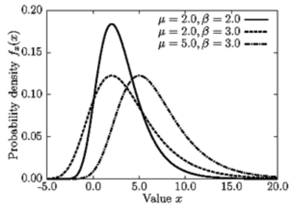
Name	**Image**	**Description**
Uniform distribution	 $a \le x \le b,\ a < b$	Distribution parameter • Lower bound *a* • Upper bound *b* > *a* Mean value and standard deviation • $\overline{X}=\frac{1}{2}(a+b)$ • $\sigma_X=\frac{1}{\sqrt{12}}(b-a)$ PDF and CDF • $f_X(x)=\frac{1}{b-a};\ a\le x\le b$ • $F_X(x)=\frac{x-a}{b-a};\ a\le x\le b$
Multi-uniform distribution	 $x_1 \le x \le x_n,\ x_{i+1}$ $p_i \le p_{i+1},\ p_1=0,\ p_n=1$	Distribution parameter • *n* pairs of: • Value $x_i < x_{i+1}$ • $p_i \le p_{i+1},\ p_1=0,\ p_n=1$ Mean value and standard deviation • $\overline{X}=\frac{1}{2}\sum_{i=1}^{n-1} g_i(x_{i+1}^2-x_i^2)$ • $\sigma_X=\sqrt{\frac{1}{3}\sum_{i=1}^{n-1} g_i(x_{i+1}^3-x_i^3)-\bar{X}^2}$ • $g_i=(p_{i+1}-p_i)/(x_{i+1}-x_i)$ PDF and CDF • $f_X(x)=g_i,\ x_i\le x<x_{i+1}$ • $F_X(x)=g_i(x-x_i)+p_i$

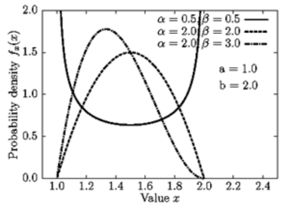
Name	**Image**	**Description**
Log-uniform distribution	 $a \leq x \leq b,\ 0 < a < b$	Distribution parameter • Lower bound $a > 0$ • Upper bound $b > a$ Mean value and standard deviation • $\overline{X} = \frac{b-a}{\ln(b)-\ln(a)}$ • $\sigma_X = \sqrt{\frac{1}{2}\overline{X}(a+b)-\bar{X}^2}$ PDF and CDF • $f_X(x) = \frac{1}{x(\ln(b)-\ln(a))};\ a \leq x \leq b$ • $F_X(x) = \frac{\ln(x)-\ln(a)}{\ln(b)-\ln(a)};\ a \leq x \leq b$
Normal distribution	 $-\infty < x < \infty,\ s > 0$	Distribution parameter • Location parameter μ • Scale parameter $s > 0$ Mean value and standard deviation • $\overline{X} = \mu$ • $\sigma_X = s$ PDF and CDF • $f_X(x) = \phi_{u,s}(x) = \frac{1}{\sqrt{2\pi s^2}} \exp\frac{-(x-\mu)^2}{2s^2}$ • $F_X(x) = \Phi_{u,s}(x) = \int_{-\infty}^{x} \phi_{\mu,s}(z)dz$

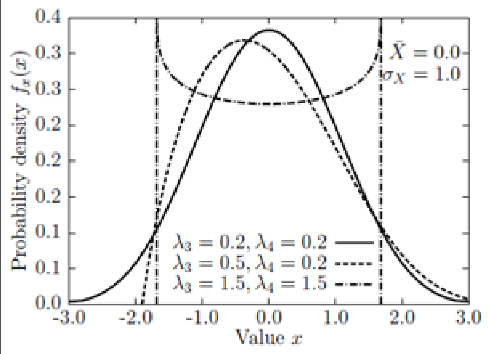
Release 2021 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Name	Image	Description
Truncated normal distribution	 $a \le x \le b,\ s > 0,\ a < b$	Distribution parameter • Location parameter μ • Scale parameter $s > 0$ • Lower bound a • Upper bound b Mean value and standard deviation • $\overline{X} = \mu + s\dfrac{\phi(\alpha) - \phi(\beta)}{\Phi(\beta) - \Phi(\alpha)}$ • $\sigma_X = s\sqrt{1 + \dfrac{\alpha\phi(\alpha) - \beta\phi(\beta)}{\Phi(\beta) - \Phi(\alpha)} - \left(\dfrac{\phi(\alpha) - \phi(\beta)}{\Phi(\beta) - \Phi(\alpha)}\right)^2}$ PDF and CDF • $f_X(x) = \dfrac{\phi(\xi)}{s(\Phi(\beta) - \Phi(\alpha))}\ \ a \le x \le b$ • $F_X(x) = \dfrac{\Phi(\xi) - \Phi(a)}{\Phi(\beta) - \Phi(\alpha)}\ \ a \le x \le b$ • $\alpha = \dfrac{a - \mu}{s}, \quad \beta = \dfrac{b - \mu}{s}, \quad \xi = \dfrac{x - \mu}{s}$ • $\varphi(\cdot) = \varphi_{0,1}(\cdot), \quad \Phi(\cdot) = \Phi_{0,1}(\cdot)$
Log-normal distribution	 $0 < x < \infty; s > 0$	Distribution parameter • Normal location parameter μ • Normal scale parameter $s > 0$ Mean value and standard deviation • $\overline{X} = \exp(\mu + \dfrac{s^2}{2})$ • $\sigma_X = \overline{X}\sqrt{\exp(s^2) - 1}$ PDF and CDF • $f_X(x) = \dfrac{1}{x}\phi_{\mu,s}(\log x),\ x > 0$ • $F_X(x) = \dfrac{1}{x}\Phi_{\mu,s}(\log x),\ x > 0$

Name	Image	Description
Triangular distribution	 $a \le x \le b,\ a<c<b$	Distribution parameter • Lower bound a • Mode value $c > a$ • Upper bound $b > c$ Mean value and standard deviation • $\overline{X}=\frac{1}{3}(a+b+c)$ • $\sigma_X=\frac{1}{\sqrt{18}}\sqrt{a^2+b^2+c^2-ab-ac-bc}$ PDF and CDF • $f_X(x)=\begin{cases}\frac{2(x-a)}{(b-a)(c-a)} & \text{if } a\le x\le c \\ \frac{2(b-x)}{(b-a)(b-c)} & \text{if } c<x\le b\end{cases}$ • $F_X(x)=\begin{cases}\frac{(x-a)^2}{(b-a)(c-a)} & \text{if } a\le x\le c \\ 1-\frac{(b-x)^2}{(b-a)(b-c)} & \text{if } c<x\le b\end{cases}$
Exponential distribution	 $\epsilon \le x \le \infty,\ \beta>0$	Distribution parameter • Scale parameter $\beta>0$ • Shift parameter Mean value and standard deviation • $\overline{X}=\beta+\epsilon$ • $\sigma_X=\beta$ PDF and CDF • $f_X(x)=\frac{1}{\beta}\exp\left(-\frac{x-\epsilon}{\beta}\right),\ x\ge\epsilon$ • $F_X(x)=1-\exp\left(-\frac{x-\epsilon}{\beta}\right),\ x\ge\epsilon$

Name	**Image**	**Description**
Weibull distribution	 $0 \le x \le \infty,\ \lambda, k > 0$	Distribution parameter • Scale parameter $\lambda > 0$ • Shape parameter $k > 0$ Mean value and standard deviation • $\overline{X}=\lambda\Gamma(1+\frac{1}{k})$ • $\sigma_X=\lambda\sqrt{(\Gamma(1+\frac{2}{k})-(\Gamma(1+\frac{1}{k}))^2}$ PDF and CDF • $f_X(x)=\frac{k}{\lambda}(\frac{x}{\lambda})^{k-1}\exp(-(\frac{x}{\lambda})^k),\ x\ge 0$ • $F_X(x)=1-\exp(-(\frac{x}{\lambda})^k),\ x\ge 0$ • $\Gamma(\cdot)$ is the gamma function
Frechet distribution	 $0 < x < \infty;\ s > 0;\ \alpha > 2$	Distribution parameter • Scale parameter $s > 0$ • Shape parameter $\alpha > 2$ Mean value and standard deviation • $\overline{X}=s\Gamma(1-\frac{1}{\alpha})$ • $\sigma_X=s\sqrt{(\Gamma(1+\frac{2}{\alpha})-(\Gamma(1+\frac{1}{\alpha}))^2}$ PDF and CDF • $f_X(x)=\frac{\alpha}{s}(\frac{x}{s})^{-1-\alpha}\exp(-(\frac{x}{s})^{-\alpha}),\ x>0$ • $F_X(x)=\exp(-(\frac{x}{s})^{-\alpha}),\ x>0$
Gumbel distribution	 $-\infty < x < \infty,\ \beta > 0$	Distribution parameter • Location parameter μ • Scale parameter $\beta > 0$ Mean value and standard deviation

Name	Image	Description
		• $\overline{X}=\mu+\gamma\beta$ • $\sigma_X=\frac{\pi}{\sqrt{6}}\beta$ PDF and CDF • $f_X(x)=\frac{1}{\beta}\exp(-z-\exp(-z))$ • $F_X(x)=\exp(-\exp(-z))$ • $z=\frac{x-\mu}{\beta}$ • γ is the is the Euler-Mascheroni constant
Beta distribution	 $a \leq x \leq b,\ a<b,\ \alpha,\beta>0$	Distribution parameter • Lower bound *a* • Upper bound *b* • Shape parameter $\alpha > 0$ • Shape parameter $\beta > 0$ Mean value and standard deviation • $\overline{X}=\frac{\alpha b-\beta a}{\alpha+\beta}$ • $\sigma_X=(b-a)\sqrt{\frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}}$ PDF and CDF • $f_X(x)=\frac{u^{\alpha-1}(1-u^{\beta-1})}{B(\alpha,\beta)(b-a)}$ • $F_X(x)=\frac{B(u,\alpha,\beta)}{B(\alpha,\beta)}$ • $u=\frac{x-a}{b-a}$ • $B(\cdot,\cdot)$ is the Beta function • $B(u,\cdot,\cdot)$ is the incomplete Beta function

Name	**Image**	**Description**
Lambda distribution	 $\lambda_2 > 0,\ \lambda_3 \lambda_4 > -0.25,\ \lambda_3 \lambda_4 \neq 0$	Distribution parameter • Location parameter λ_1 • Scale parameter $\lambda_2 > 0$ • Shape parameter $\lambda_3 > -0.25, \lambda_3 \neq 0$ • Shape parameter $\lambda_4 > -0.25, \lambda_4 \neq 0$ Mean value and standard deviation • $\overline{X}=\lambda_1-\frac{1}{\lambda_2}\left(\frac{1}{\lambda_3+1}-\frac{1}{\lambda_4+1}\right)$ • $\sigma_X=\sqrt{\frac{b-a^2}{\lambda^2}}$ • $a=\frac{1}{\lambda_3(1+\lambda_3)}-\frac{1}{\lambda_4(1+\lambda_4)}$ • $b=\frac{1}{\lambda_3^2(1+2\lambda_3)}+\frac{1}{\lambda_4^2(1+2\lambda_4)}-\frac{2}{\lambda_3\lambda_4 B(1+\lambda_3,1+\lambda_4)}$ • $B(\cdot,\cdot)$ is the Beta function PDF and inverse CDF • $f_X(u)=\lambda_2\left[\frac{u^{\lambda 3^{-1}}}{\lambda_3}+\frac{(1-u)^{\lambda 4^{-1}}}{\lambda_4}\right]^{-1}$ • $F_X^{-1}(u)=\lambda_1+\frac{1}{\lambda_2}\left[\frac{u^{\lambda 3^{-1}}}{\lambda_3}-\frac{(1-u)^{\lambda 4^{-1}}}{\lambda_4}\right]$ • $u=F_X(x)$ • CDF is not available in closed form

Continuous distribution types: distribution parameters, probability density function (PDF) $f_X(x)$, cumulative distribution function (CDF) $F_X(x)$, mean value $\overline{X}$, and standard deveation σ_X.

References

[1] Bäck, T. (1996). Evolution strategies: An alternative evolutionary algorithm. *Lecture Notes in Computer Science 1063/1996*. 1-20.

[2] Bjerager, P. (1988). Probability integration by directional simulation. *Journal of Engineering Mechanics, ASCE 114*. 1285-1302.

[3] Bourgund, U. and C. Bucher. (1986). Importance sampling procedure using design points. Technical report. Institut für Mechanik, Universität Innsbruck.

[4] Branke, J., K. Deb, K. Miettinen, and R. Slowinski. (2008). *Multiobjective Optimization*. Springer.

[5] Bucher, C. (2009). *Computational Analysis of Randomness in Structural Mechanics*. London: CRC Press, Taylor & Francis Group.

[6] Bucher, C. and U. Bourgund. (1990). A fast and efficient response surface approach for structural reliability problems. *Structural Safety 7*. 57-66.

[7] Deák, I. (1980). Three digit accurate multiple normal probabilities. *Numerische Mathematik 35*. 369-380.

[8] Engelbrecht, A. P. (2005). *Fundamentals of Computational Swarm Intelligence*. John Wiley & Sons.

[9] Etman, L., J. Adriaens, M. van Slagmaat, and A. Schoofs. (1996). Crashworthiness design optimization using multipoint sequential linear programming. *Structural Optimization 12*. 222-228.

[10] Eurocode EN 1990. Basis of structural design.

[11] Faravelli, L. (1989). Response surface approach for reliability analysis. *Journal of Engineering Mechanics, ASCE 115*. 2763-2781.

[12] Fogel, L. J., A.J. Owens, and M.J. Walsh. (1966). *Artificial Intelligence through Simulated Evolution*. John Wiley & Sons.

[13] Holland, J.H. (1975). *Adaptation in Natural and Artificial Systems*. University of Michigan Press.

[14] Hungtington, D. E. and C. S. Lyrintzis. (1998). Improvements to and limitations of Latin hypercube sampling. *Probabilistic Enginerring Mechanics 13*. 245-253.

[15] Iman, R. L. and W. J. Conover. (1982). A distribution-free approach to inducing rank correlation among input variables. *Communications in Statistics - Simulation and Computation 11*. 311-334.

[16] Kelley, C.T. (1999). *Iterative Methods for Optimization*. Philadelphia: Siam, Society for Industrial and Applied Mathematics.

[17] Kennedy, J. and R. Eberhart. (1995). Particle swarm optimization. *Proceedings of the IEEE International Conference on Neural Networks*. Volume 4.

[18] Lancaster, P. and K. Salkauskas. (1981). Surface generated by moving least squares methods. *Mathematics of Computation 37*. 141-158.

[19] H. Madsen, S. Krenk, and N. Lind. (1986). *Methods of Structural Safety*. Englewood Cliffs, NJ, USA: Prentice Hall.

[20] MATLAB. (2010). *User's guide*. (Version 7.11.0 (R2010b) ed.). Natick, Massachusetts: Math-Works Inc.

[21] McKay, M., R. Beckman, and W. Conover. (1979). A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics 21*. 239-245.

[22] Montgomery, D. C. and G. C. Runger. (2003). *Applied Statistics and Probability for Engineers*. (Third ed.). John Wiley & Sons.

[23] Most, T. and J. Will. (2008). Metamodel of Optimal Prognosis - an automatic approach for variable reduction and optimal metamodel selection. *Proc. Weimarer Optimierungs- und Stochastiktage 5.0, Weimar, Germany, November 20-21, 2008*.

[24] Myers, R. and D.C. Montgomery. (2002). *Response Surface Methodology*. (2 ed.). John Wiley & Sons, Inc.

[25] Nataf, A. Détermination des distributions de probabilités dont les marges sont données. *Comptes Rendus de l'Academie des Sciences 225*. 42-43.

[26] Poli, R., J. Kennedy, and T. Blackwell. (2007). Particle swarm optimization, An overview. *Swarm Intelligence 1*. 33-57.

[27] Rechenberg, I. (1964). Kybernetische Lösungsansteuerung einer experimentellen Forschungsaufgabe. *Annual Conference of the WGLR, Berlin, 1964*.

[28] Riedel, J., S. Blum, R. Puisa, and M. Wintermantel. (2005). Adaptive mutation strategies for evolutionary algorithms: A comparative benchmark study. *Proc. Weimarer Optimierungs- und Stochastiktage 2.0, Weimar, Germany*.

[29] Roos, D., T. Most, J. F. Unger, and J. Will. (2007). Advanced surrogate models within the robustness evaluation. *Proc. Weimarer Optimierungs- und Stochastiktage 4.0, Weimar, Germany, November 29-30, 2007*.

[30] Rubinstein, R. Y. (1981). *Simulation and the Monte Carlo Method*. New York: John Wiley & Sons.

[31] Saltelli, A. et al. (2008). *Global Sensitivity Analysis. The Primer*. Chichester, England: John Wiley & Sons, Ltd.

[32] Schittkowski, K. (1986). NLPQL: A Fortran subroutine for solving constrained nonlinear programming problems. *Annals of Operations Research 5*. 485-500.

[33] Stander, N. and K. Graig. (2002). On the robustness of a simple domain reduction scheme for simulation-based optimization. *Engineering Computations 19*. 431-450.

[34] Zitzler, E. , M. Laumanns, and L. Thiele. (2001). SPEA2: Improving the Strength Pareto Evolutionary Algorithm. Technical Report 103. Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH) Zurich.

Release 2021 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.